Problem 1 Write the best **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   string s; char c = 'A'; double d = 1.1;
   int a[4] = {3, 1, 4, 2};
   bool b[2][3] = {{true, false, true}, {false, true, true}};

   s = asString(c); cout << s << endl;      // (a) prints: A
   doubleIt(d); cout << d << endl;          // (b) prints: 2.2
   doubleThem(a, 4);  cout << a[0] << endl; // (c) prints 6
   printArray(b, 2, 3);                     // (d) prints TFT FTT
   c = randomLetter(); cout << c << endl;   // (e) prints a random letter eg Z
   return 0;
}
```

(a) Title line for **asString**.

Answer:

```
string asString(char x)
```

(b) Title line for **doubleIt**.

Answer:

```
void doubleIt(double &x)
```

(c) Title line for **doubleThem**.

Answer:

```
void doubleThem(int x[], int cap)
```

(d) Title line for **printArray**.

Answer:

```
void printArray(bool x[][3], int r, int c)
```

(e) Title line for **randomLetter**.

Answer:

```
char randomLetter()
```

Problem 2 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double down(int x[], int cap, int gap) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= gap)
     ans += x[i];
  return ans / 10;
}

int main() {
    int x[4] = {3, 1, 4, 1};
    cout << x[2] << endl;                    // line (a)
    cout << x[5/3] << endl;                  // line (b)
    cout << down(x, 4, 1) << endl;           // line (c)
    cout << down(x, 4, 3) << endl;           // line (d)
    cout << down(x, x[0], x[x[1]]) << endl;  // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
4
```

(b) What is the output at line (b)?
Answer:

```
1
```

(c) What is the output at line (c)?
Answer:

```
0.9
```

(d) What is the output at line (d)?
Answer:

```
0.4
```

(e) What is the output at line (e)?
Answer:

```
0.8
```

Problem 3 Write a function called *diff2* that returns the absolute value of the difference of the first two digits of a positive integer parameter. If the parameter has just one digit, that digit should be returned.
For example, a program that uses the function *diff2* follows.

```
int main() {
   cout << diff2(7070);        // prints 7
   cout << endl;
   cout << diff2(7907);        // prints 2
   cout << endl;
   cout << diff2(7);           // prints 7
   cout << endl;
   return 0;
}
```

Answer:

```
int diff2(int x) {
   if (x < 100) {
      int ans = x / 10 - x % 10;
      if (ans < 0) ans = -ans;
      return ans;
   }
   return diff2( x / 10);
}
```

Problem 4 Write a function called *evenLessOdd* that returns the sum of the even valued entries minus the sum of the odd valued entries in an array of integers.

For example, a program that uses the function *evenLessOdd* follows. The first output is $2 = 8 - 1 - 5$ and the second is $-10 = -1 - 1 - 5 - 3$.

```
int main() {
   int x[3] = {8, 1, 5};
   int y[4] = {1, 1, 5, 3};
   cout << evenLessOdd(x, 3) << endl;       // prints 2
   cout << evenLessOdd(y, 4) << endl;       // prints -10
   return 0;
}
```

Answer:

```
int evenLessOdd(int a[], int cap) {
   int answer = 0;
   for (int i = 0; i < cap; i++)
      if ((a[i] % 2) == 0) answer += a[i];
      else answer -= a[i];
   return answer;
}
```

Problem 5 Write the best **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   string s; char c = 'A'; double d = 1.1;
   int a[4] = {3, 1, 4, 2};
   bool b[2][3] = {{true, false, true}, {false, true, true}};

   d = randomNumber(); cout << d << endl; // (a) prints a random number eg 1.5
   printThem(a, 4);                                          // (b) prints 3142
   b[1][0] = majority(b, 2, 3); if (b[1][0]) cout << "true\n"; // (c) prints true
   doubleIt(a[1]); cout << a[1] << endl;                     // (d) prints: 2
   s = asString(b[0][0]); cout << s << endl;                 // (e) prints: True
   return 0;
}
```

(a) Title line for **randomNumber**.

Answer:

```
double randomNumber()
```

(b) Title line for **printThem**.

Answer:

```
void printThem(int x[], int cap)
```

(c) Title line for **majority**.

Answer:

```
bool majority(bool x[][3], int r, int c)
```

(d) Title line for **doubleIt**.

Answer:

```
void doubleIt(int &x)
```

(e) Title line for **asString**.

Answer:

```
string asString(bool x)
```

Problem 6 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double down(int x[], int cap, int &gap) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= gap)
     ans += x[i];
  gap += 2;
  return ans / 10;
}

int main() {
    int x[4] = {3, 2, 1, 8};
    int a = 4, b = 1;
    cout << x[7/3] << endl;                    // line (a)
    cout << down(x, a, b) << endl;             // line (b)
    cout << down(x, a, b) << endl;             // line (c)
    cout << down(x, x[0], x[x[2]]) << endl;    // line (d)
    cout << x[2] << endl;                      // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
1
```

(b) What is the output at line (b)?

Answer:

```
1.4
```

(c) What is the output at line (c)?

Answer:

```
1.1
```

(d) What is the output at line (d)?

Answer:

```
0.4
```

(e) What is the output at line (e)?

Answer:

```
1
```

Problem 7 Write a function called *unlucky* that returns an answer of *true* if the first two digits of a positive integer parameter add to 13. Otherwise it returns *false*. (It returns *false* if the parameter has fewer than 2 digits.) For example, a program that uses the function *unlucky* follows.

```
int main() {
   if (unlucky(6789)) cout << "Unlucky!\n";  // prints Unlucky!
   if (unlucky(6889)) cout << "Unlucky!\n";  // prints
   if (unlucky(6)) cout <<    "Unlucky!\n";  // prints
   if (unlucky(49)) cout <<   "Unlucky!\n";  // prints Unlucky!
   return 0;
}
```

Answer:

```
bool unlucky(int x) {
   if (x < 100)
      return x / 10 + x % 10 == 13;
   return unlucky( x / 10);
}
```

Problem 8 Write a function called *lastOdd* that returns the last odd valued entry in an array or returns 0 if there is no odd value.

For example,

```
int main() {
   int x[3] = {8, 1, 7};
   int y[5] = {1, 2, 5, 4, 6};
   int z[2] = {2, 2};
   cout << lastOdd(x, 3) << endl;       // prints 7
   cout << lastOdd(y, 5) << endl;       // prints 5
   cout << lastOdd(z, 2) << endl;       // prints 0
   return 0;
}
```

Answer:

```
int lastOdd(int a[], int cap) {
   for (int i = cap - 1; i >= 0; i--)
      if ((a[i] % 2) == 1) return a[i];
   return 0;
}
```

Problem 9 Write the best **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   string s; char c = 'A'; double d = 4.0;
   int a[4] = {3, 1, 4, 2};
   bool b[2][3] = {{true, false, true}, {false, true, true}};

   printThem(b, 2, 3);                          // (a) prints TFT FTT
   fixLies(b, 2, 3); printThem(b, 2, 3);        // (b) prints FTF TFF
   d = cubeIt(d); cout << d << endl;            // (c) prints: 64.0
   cubeInt(a[2]); cout << a[2] << endl;         // (d) prints: 64
   a[1] = reverseDigits(a[2]); cout << a[1] << endl; // (e) prints: 1
   return 0;
}
```

(a) Title line for **printThem**.

Answer:

```
void printThem(bool x[][3], int r, int c)
```

(b) Title line for **fixLies**.
Answer:

```
void fixLies(bool x[][3], int r, int c)
```

(c) Title line for **cubeIt**.
Answer:

```
double cubeIt(double x)
```

(d) Title line for **cubeInt**.
Answer:

```
void cubeInt(int &x)
```

(e) Title line for **reverseDigits**.
Answer:

```
int reverseDigits(int x)
```

Problem 10 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double down(int x[], int cap, int &gap) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= gap)
     ans += x[i];
  gap += 2;
  return ans / 10;
}

int main() {
    int x[4] = {9, 1, 3, 2};
    int a = 4, b = 2;
    cout << x[9/3] << endl;                    // line (a)
    cout << down(x, a, b) << endl;             // line (b)
    cout << down(x, a, b) << endl;             // line (c)
    cout << down(x, x[2], x[x[2]]) << endl;    // line (d)
    cout << x[3] << endl;                      // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
2
```

(b) What is the output at line (b)?
Answer:

```
1.2
```

(c) What is the output at line (c)?
Answer:

```
0.9
```

(d) What is the output at line (d)?

Answer:

```
1.2
```

(e) What is the output at line (e)?

Answer:

```
4
```

Problem 11 Write a function called *add7* that returns an answer found by putting a 7 in front of the first digit of a positive integer.

For example, a program that uses the function *add7* follows.

```
int main() {
   cout << add7(1) << "\n";    // prints 71
   cout << add7(17) << "\n";   // prints 717
   cout << add7(456) << "\n";  // prints 7456
   return 0;
}
```

Answer:

```
int add7(int x) {
   if (x == 0) return 7;
   return add7( x / 10) * 10 + x % 10;
}
```

Problem 12 Write a function called *indexFirstOdd* that returns the index of the first odd valued entry in an array or returns -1 if there is no odd value. (The index of an entry is its position in the array.)

For example,

```
int main() {
   int x[3] = {8, 8, 7};
   int y[5] = {7, 2, 5, 1, 9};
   int z[2] = {2, 2};
   cout << indexFirstOdd(x, 3) << endl;       // prints 2
   cout << indexFirstOdd(y, 5) << endl;       // prints 0
   cout << indexFirstOdd(z, 2) << endl;       // prints -1
   return 0;
}
```

Answer:

```
int indexFirstOdd(int a[], int cap) {
   for (int i = 0; i < cap; i++)
      if ((a[i] % 2) == 1) return i;
   return -1;
}
```

Problem 13 Write the best **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   string fullName = "Freddy Next Door";
   int a2[2][3] = {{-2, 4, 3}, {-3, 4, 2}};
   int a[5] = {7, 6, 5, 9, 7};
   cout << middleDigit(19683) + 1   << endl;   // (a) prints: 7 as 6 + 1
   cout << random(a2, 2, 3) << endl;           // (b) prints random entry eg 4
   cout << initials(fullName) << endl;         // (c) prints: F.N.D.
   makePositive(a2[0][0]);                     // (d) make a2[0][0] positive
   cout << number7s(a, 5);                     // (e) prints 2: the number of 7s
   return 0;
}
```

(a) Title line for **middleDigit**.

Answer:

```
int middleDigit(int x)
```

(b) Title line for **random**.

Answer:

```
int random(int [][3], int r, int c)
```

(c) Title line for **initials**.

Answer:

```
string initials(string x)
```

(d) Title line for **makePositive**.

Answer:

```
void makePositive(int &x)
```

(e) Title line for **number7s**.

Answer:

```
int number7s(int x[], int cap)
```

Problem 14 Write the best **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   string fullName = "Freddy Next Door";
   int a2[2][3] = {{-2, 4, 3}, {-3, 4, 2}};
   int a[5] = {7, 6, 5, 9, 7};
   cout << firstLetter(fullName) << endl;      // (a) prints: F
   cout << sumFirstCol(a2, 2, 3) << endl;      // (b) prints: -5  (as -2 + - 3).
   cout << middleName(fullName) << endl;       // (c) prints: Next
   makeRandom(a2, 2, 3);                       // (d) reset the array with random entries
   cout << round(((double) a[0])/((double) a[1])); // (e) prints 1
                                        //  the nearest integer to the ratio.
   return 0;
}
```

(a) Title line for **firstLetter**.

Answer:

```
char firstLetter(string x)
```

(b) Title line for **sumFirstCol**.
Answer:

```
int sumFirstCol(int [][3], int r, int c)
```

(c) Title line for **middleName**.
Answer:

```
string middleName(string x)
```

(d) Title line for **makeRandom**.
Answer:

```
void makeRandom(int x[][3], int r, int c)
```

(e) Title line for **round**.
Answer:

```
int round(double x)
```

Problem 15 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
    x = x + 1;
    y = y - 1;
    return y;
}

int main() {
    int x = 2, y = 7, z = 10;   string s = "007";
    cout << ((double) y) / x << endl;         // line (a)
    if (!((x > y) && (y > 5))) s = "008";
    cout << s << endl;                        // line (b)
    z %= y; cout << z << endl;                // line (c)
    cout << fun(z, y) << endl;                // line (d)
    fun(x, y); cout << y - x * 2 << endl;     // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
3.5
```

(b) What is the output at line (b)?
Answer:

```
008
```

(c) What is the output at line (c)?
Answer:

```
3
```

(d) What is the output at line (d)?
Answer:

```
6
```

(e) What is the output at line (e)?
Answer:

```
1
```

Problem 16 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x, int &y) {
    x = x + 1;
    y = y - 1;
    return y;
}

int main() {
    int x = 3, y = 9, z = 10;   string s = "Yes";
    cout << ((double) x) / z << endl;          // line (a)
    if (!((x > y) || (y > 5))) s = "No";
    cout << s << endl;                         // line (b)
    z %= y; cout << z << endl;                 // line (c)
    cout << fun(z, y) << endl;                 // line (d)
    fun(x, y); cout << y - x % 2 << endl;      // line (e)
}
```

(a) What is the output at line (a)?
Answer:

```
0.3
```

(b) What is the output at line (b)?
Answer:

```
Yes
```

(c) What is the output at line (c)?
Answer:

```
1
```

(d) What is the output at line (d)?
Answer:

```
8
```

(e) What is the output at line (e)?
Answer:

```
6
```

Problem 17 Write a function called *removeLast0* that prints an integer parameter without its rightmost 0. If there is no 0, print the number itself. If the number is 0, print nothing.

For example, a program that uses the function *removeLast0* follows.

```
int main() {
   removeLast0(7070);        // prints 707
   cout << endl;
   removeLast0(7007);        // prints 707
   cout << endl;
   removeLast0(777);         // prints 777
   cout << endl;
   return 0;
}
```

Answer:

```
void removeLast0(int n) {
   if (n == 0) return;
   if (n% 10 == 0) cout << n/10;
   else {
       removeLast0(n/10);
       cout << n % 10;
   }
}
```

Problem 18 Write a function called *removeLast7* that removes the rightmost 7 from an integer parameter. If there is no 7, it makes no change.

For example, a program that uses the function *removeLast7* follows.

```
int main() {
   cout << removeLast7(777)  << endl;       // prints 77
   cout << removeLast7(1727) << endl;       // prints 172
   cout << removeLast7(1234) << endl;       // prints 1234
   return 0;
}
```

Answer:

```
int removeLast7(int n) {
   if (n == 0) return 0;
   if (n % 10 == 7) return n/10;
   return 10 * removeLast7(n/10) + n%10;
}
```

Problem 19 Write a function called *largestGap* that returns the largest gap between two adjacent elements of an array.

For example, a program that uses the function *largestGap* follows, it prints 7 since the largest gap is between the 9 and the 2.

```
int main() {
   int x[] = {3, 1, 4, 1, 5, 9, 2, 6};
   cout << largestGap(x, 8) << endl;  // prints 7
   return 0;
}
```

Answer:

```
int largestGap(int x[], int n) {
   int max = x[0] - x[1];
   for (int i = 1; i < n; i++) {
      if (x[i] - x[i - 1] > max) max = x[i] - x[i - 1];
      if (x[i - 1] - x[i] > max) max = x[i - 1] - x[i];
   }
   return max;
}
```

Problem 20 Write a function called *smallestProduct* that returns the smallest product formed by two adjacent elements of an array.

For example, a program that uses the function *smallestProduct* follows, it prints 3 since the smallest product is between the 3 and the 1.

```
int main() {
   int x[] = {3, 1, 4, 1, 5, 9, 2, 6};
   cout << smallestProduct(x, 8) << endl;  // prints 3
   return 0;
}
```

Answer:

```
int smallestProduct(int x[], int n) {
   int min = x[0] * x[1];
   for (int i = 1; i < n; i++)
      if (x[i] * x[i - 1] < min) min = x[i] * x[i - 1];
   return min;
}
```

Problem 21 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};

   x = larger(x + y, z);               // (a) sets x as the larger
   x = largest(x, y, y, z);            // (b) sets x as the largest
   printAll(b, x, y);                  // (c) print them all
   boost(x, y);                        // (d) increase x by the value of y
   boost(y, mystery(y, z));            // (e) boosts y by a mystery amount
   return 0;
}
```

(a) Title line for **larger**.

Answer:

```
int larger(int a, int b)
```

(b) Title line for **largest**.

Answer:

```
int largest(int a, int b, int c, int d)
```

(c) Title line for **printAll**.

Answer:

```
void printAll(double a[], int b, int c)
```

(d) Title line for **boost**.

Answer:

```
void boost(int &a, int b)
```

(e) Title line for **mystery**.

Answer:

```
int mystery(int a, int b)
```

Problem 22 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};

   larger(x + y, z);                       // (a) prints the larger
   x = middle(x, y, y, z, z);              // (b) sets x as the middle value
   printAll(sqrt(b[1]), rand());           // (c) print them all
   swap(x, y);                             // (d) swap them
   cout << mystery(y, mystery(y, b[0]));   // (e) a mystery function
   return 0;
}
```

(a) Title line for **larger**.

Answer:

```
void larger(int a, int b)
```

(b) Title line for **middle**.

Answer:

```
int middle(int a, int b, int c, int d, int e)
```

(c) Title line for **printAll**.

Answer:

```
void printAll(double a, int b)
```

(d) Title line for **swap**.

Answer:

```
void swap(int &a, int &b)
```

(e) Title line for **mystery**.

Answer:

```
double mystery(int a, double b)
```

Problem 23 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the sum of 3 things, here 6
   cout << sum3(1,3,c) << endl;
 // (b) Prints decimal form of fraction b/c, here 0.5
   cout << fraction(b, c) << endl;
 // (c) Fill array with random integers
   randomFill(a, 4);
 // (d) Print array backwards, with entries separated by spaces
   backPrint(a, 4);
 // (e) Print the first digit, assume argument is positive.  Here 1.
   firstDigit(19683);
   cout << endl;
   return 0;
}
```

Answer:

(a)

```
int sum3(int x, int y, int z) {
    return x + y + z;
}
```

(b)

```
double fraction (int x, int y) {
    return ((double) x) / y;
}
```

(c)

```
void randomFill(int x[], int cap) {
   for (int i = 0; i < cap; i++) x[i] = rand();
}
```

(d)

```
void backPrint(int x[], int cap) {
   for (int i = cap - 1; i >= 0; i--)
       cout << x[i] << " ";
   cout << endl;
}
```

(e)

```
void firstDigit(int x) {
   if (x < 10) cout << x;
   else firstDigit(x / 10);
}
```

Problem 24 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the average of 3 things, here 2.0
   cout << average3(1,3,c) << endl;
 // (b) Print the fraction b/c as a percentage, here 50.0%
   cout << percentage(b, c) << "%" << endl;
 // (c) Fill array with random positive single digit integers
   randomFill(a, 4);
 // (d) Print array, with entries separated by spaces
   print(a, 4);
 // (e) Print the second digit, assume argument is at least 10.  Here print 9.
   cout << secondDigit(19683) << endl;
   return 0;
}
```

Answer:

(a)

```
double average3(int x, int y, int z) {
    return (x + y + z) / 3.0;
}
```

(b)

```
double percentage(int x, int y) {
    return  100.0 * x / y;
}
```

(c)

```
void randomFill(int x[], int cap) {
   for (int i = 0; i < cap; i++)
      x[i] = rand() % 9 + 1;
}
```

(d)

```
void print(int x[], int cap) {
   for (int i = 0; i < cap; i++)
       cout << x[i] << " ";
   cout << endl;
}
```

(e)

```
int secondDigit(int x) {
   if (x < 100) return x % 10;
   else return secondDigit(x / 10);
}
```

Problem 25 Write a function called *gcd* that returns the greatest common divisor of two positive integers. For example, a program that uses the function *gcd* follows.

```
int main() {
   cout << gcd(10, 15) << endl;       // prints 5
   cout << gcd(11, 15) << endl;       // prints 1
   cout << gcd(0, 15) << endl;        // prints 15
   return 0;
}
```

Answer:

```
int gcd(int x, int y) {
   if (y == 0) return x;
   return gcd(y, x % y);
}
```

Problem 26 Write a function called *removeFirst* that removes the first digit of a positive integer and returns the result (or returns 0 if the integer has only one digit).

For example, a program that uses the function *removeFirst* follows.

```
int main() {
   cout << removeFirst(19683) << endl;    // prints 9683
   cout << removeFirst(11) << endl;       // prints 1
   cout << removeFirst(1) << endl;        // prints 0
   return 0;
}
```

Answer:

```
int removeFirst(int x) {
   if (x < 10) return 0;
   return 10 * removeFirst(x/10) + x % 10;
}
```

Problem 27 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter to enter 250 integers.
2. It computes the average of the 250 integers that the user supplies.
3. It prints out exactly those numbers entered by the user that differ from the average by no more than 10.

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int data[250];
   int count = 250;

   cout << "Enter 250 integers: ";
   for (int i = 0; i < count; i++) cin >> data[i];

   int sum = 0;
   for (int i = 0; i < count; i++) sum = sum + data[i];
   double average = sum / ((double) count);

   for (int i = 0; i < count; i++)
     if ((average - data[i]) <= 10.0 && (data[i] - average) <= 10.0)
        cout << data[i] << endl;

   return 0;
}
```

Problem 28 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter to enter 250 integers.

2. It prints out exactly the negative numbers entered by the user in the reverse of their order of input.

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int data[250];
   int count = 250;

   cout << "Enter 250 integers: ";
   for (int i = 0; i < count; i++) cin >> data[i];

   for (int i = count - 1; i >= 0; i--)
     if (data[i] < 0)
        cout << data[i] << endl;

   return 0;
}
```

Problem 29 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4;

   cout << max(a, 4) << endl;              // (a) prints: 358
   reverse(a, 4);                          // (b) prints: 358 265 315 314
   b = add(b, c);                          // (c) b becomes 5
   cout << difference(a[0], a[1]) << endl; // (d) prints: 1
   a[0] = sum(a[1], c);                    // (e) a[0] becomes 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int x[], int cap)
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[], int cap)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 30 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4, capacity = 4;

   swap(b, c);                              // (a) swaps values of b & c
   b = last(a, 4);                          // (b) b becomes 358
   c = add(a[1], a[0]);                     // (c) c becomes 629
   cout << max(a[1], 1) << endl;            // (d) prints: 314
   cout << max(a, capacity, 700) << endl;   // (e) prints 700
   return 0;
}
```

(a) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **last**.

Answer:

```
int last(int x[], int cap)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **max**.

Answer:

```
int max(int x, int y)
```

(e) Title line for **max**.

Answer:

```
int max(int x[], int y, int z)
```

Problem 31 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4;

   cout << max(4, a) << endl;               // (a) prints: 358
   reverse(a, 4);                           // (b) a becomes 358,265,315,314
   b = add(b, b, c);                        // (c) b becomes 6
   cout << difference(a[1], 300) << endl;   // (d) prints: 15
   addOn(a[1], c);                          // (e) a[1] changes to 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int cap, int x[])
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[], int cap)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y, int z)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **addOn**.

Answer:

```
void addOn(int &x, int y)
```

Problem 32 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4, capacity = 4;

   swap(a[3], c);                          // (a) swaps values of a[3] & c
   b = first(a);                           // (b) b becomes 314
   a[3] = add(a[1], a[0]);                 // (c) a[3] becomes 629
   cout << min(a, capacity) << endl;       // (d) prints: 265
   printMin(a, capacity);                  // (e) prints: 265
   return 0;
}
```

(a) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **first**.

Answer:

```
int first(int x[])
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **min**.

Answer:

```
int min(int x[], int y)
```

(e) Title line for **printMin**.

Answer:

```
void printMin(int x[], int y)
```

Problem 33 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4;

   cout << max(a, 2, 2) << endl;                 // (a) prints: 358
   reverse(a, 2, 2);                             // (b) prints: 358 265 315 314
   b = add(b, c);                                // (c) b becomes 5
   cout << difference(a[0][0], a[0][1]) << endl; // (d) prints: 1
   a[0][0] = sum(a[0][1], c);                    // (e) a[0][0] becomes 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int x[][2], int r, int c)
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[][2], int r, int c)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 34 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4, rows = 2, cols = 2;

   swap(b, c);                                // (a) swaps values of b & c
   b = last(a, rows, cols);                   // (b) b becomes 358
   c = add(a[0][1], a[0][0]);                 // (c) c becomes 629
   cout << max(a[0][1], 1) << endl;           // (d) prints: 315
   cout << max(a, rows, cols, 700) << endl;   // (e) prints 700
   return 0;
}
```

(a) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **last**.

Answer:

```
int last(int x[][2], int r, int c)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **max**.

Answer:

```
int max(int x, int y)
```

(e) Title line for **max**.

Answer:

```
int max(int x[][2], int r, int c, int z)
```

Problem 35 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4;

   cout << max(2, 2, a) << endl;              // (a) prints: 358
   reverse(a, 2, 2);                          // (b) a becomes 358,265,315,314
   b = add(b, b, c);                          // (c) b becomes 6
   cout << difference(a[0][1], 300) << endl;  // (d) prints: 15
   addOn(a[0][1], c);                         // (e) a[0][1] changes to 319
   return 0;
}
```

(a) Title line for **max**.

Answer:

```
int max(int r, int c, int x[][2])
```

(b) Title line for **reverse**.

Answer:

```
void reverse(int x[][2], int r, int c)
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y, int z)
```

(d) Title line for **difference**.

Answer:

```
int difference(int x, int y)
```

(e) Title line for **addOn**.

Answer:

```
void addOn(int &x, int y)
```

Problem 36 Write **title lines** for the functions that are called by the following main program. **Do not supply blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4, row = 2, col = 2;

   swap(a[1][1], c);                          // (a) swaps values of a[1][1] & c
   b = first(a);                              // (b) b becomes 314
   a[1][1] = add(a[0][1], a[0][0]);           // (c) a[1][1] becomes 629
   cout << min(a, row, col) << endl;          // (d) prints: 265
   printMin(a, row, col);                     // (e) prints: 265
   return 0;
}
```

(a) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

(b) Title line for **first**.

Answer:

```
int first(int x[][2])
```

(c) Title line for **add**.

Answer:

```
int add(int x, int y)
```

(d) Title line for **min**.

Answer:

```
int min(int x[][2], int r, int c)
```

(e) Title line for **printMin**.

Answer:

```
void printMin(int x[][2], int r, int c)
```

Problem 37 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the difference (ignoring sign), here 1
   cout << absoluteDifference(7,8) << endl;
 // (b) Prints random integer in range from b to c, assume b < c
   cout << random(b, c) << endl;
 // (c) Print square root of sum of squares of arguments, here 5.0
```

```
   cout << hyp(3, 4) << endl;
 // (d) Print array backwards, here 1413
   backPrint(a, 4);
 // (e) Print the last digit, assume argument is positive.  Here 3.
   lastDigit(19683);
   return 0;
}
```

Answer:

(a)

```
int absoluteDifference(int x, int y) {
    if (x < y) return y - x;
    return x - y;
}
```

(b)

```
int random(int x, int y) {
    return rand() % (y - x + 1) + x;
}
```

(c)

```
double hyp(int x, int y) {
    return sqrt(x * x + y * y);
}
```

(d)

```
void backPrint(int x[], int cap) {
   for (int i = cap - 1; i >= 0; i--)
       cout << x[i];
   cout << endl;
}
```

(e)

```
void lastDigit(int x) {
   cout << x % 10;
}
```

Problem 38 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the max, here 8
   cout << max(7,8) << endl;
 // (b) Swaps values
   swap(b, c);
 // (c) Print ratio, here 0.75
   cout << ratio(3, 4) << endl;
 // (d) Print number of even entries, here 1
```

```
   cout << countEven(a, 4) << endl;
 // (e) Print the first digit, assume argument is positive.  Here 1.
   firstDigit(19683);
   return 0;
}
```

Answer:

(a)

```
int max(int x, int y) {
    if (x < y) return y;
    return x;
}
```

(b)

```
void swap(int &x, int &y) {
   int temp = x;
   x = y;
   y = temp;
}
```

(c)

```
double ratio(int x, int y) {
    return ((double) x) / y;
}
```

(d)

```
int countEven(int x[], int cap) {
   int ans = 0;
   for (int i = 0; i < cap; i++)
      if (x[i] % 2 == 0) ans++;
   return ans;
}
```

(e)

```
void firstDigit(int x) {
    while (x >= 10) x = x / 10;
    cout << x;
}
```

Problem 39 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the absolute value (ignore sign), here 7
   cout << absolute(-7) << endl;
 // (b) Prints a random id number with the given length, here 007 may be printed
   random(3);
 // (c) Prints the ratio as a percentage, here 12.5% for 1/8
```

```
   cout << percentage(1, 8) << "%" << endl;
 // (d) Print every second entry of the array here 34
   skipPrint(a, 4);
 // (e) Print the last two digit, assume argument is at least 10.  Here 83.
   lastTwoDigits(19683);
   return 0;
}
```

Answer:

(a)

```
int absolute(int x) {
    if (x < 0) return - x;
    return x;
}
```

(b)

```
void random(int x) {
    for (int i = 0; i < x; i++)
       cout << rand() % 10;
    cout << endl;
}
```

(c)

```
double percentage(int x, int y) {
    return 100.0 * x / y;
}
```

(d)

```
void skipPrint(int x[], int cap) {
   for (int i = 0; i < cap; i += 2)
       cout << x[i];
   cout << endl;
}
```

(e)

```
void lastTwoDigits(int x) {
   cout << x % 100;
}
```

Problem 40 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Print the number of odd arguments, here 1
   cout << numberOdd(7,8) << endl;
 // (b) Reorder arguments so that they increase, here swap them
   sort(c, b);
 // (c) Print closest integer here 4
```

```
   cout << closest(3.75) << endl;
 // (d) Print maximum entry, here 4
   cout << max(a, 4) << endl;
 // (e) Print the first digit, assume argument is positive.  Here 1.
   cout << firstDigit(19683) << endl;
   return 0;
}
```

Answer:

(a)

```
int numberOdd(int x, int y) {
    return x % 2 + y % 2;
}
```

(b)

```
void sort(int &x, int &y) {
   if (x <= y) return;
   int temp = x;
   x = y;
   y = temp;
}
```

(c)

```
int closest(double x) {
    return (int) (x + 0.5);
}
```

(d)

```
int max(int x[], int cap) {
   int ans = x[0];
   for (int i = 0; i < cap; i++)
      if (x[i] > ans) ans = x[i];
   return ans;
}
```

(e)

```
int firstDigit(int x) {
    while (x >= 10) x = x / 10;
    return x;
}
```

Problem 41 Write a function called *numEven* that returns the number of even digits in a positive integer parameter.

For example, a program that uses the function *numEven* follows.

```
int main() {
   cout << numEven(23) << endl;       // prints 1
   cout << numEven(1212) << endl;     // prints 2
   cout << numEven(777) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int numEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return numEven(x/10) + 1;
   return numEven(x/10);
}
```

Problem 42 Write a function called *lastEven* that returns the last even digit in a positive integer parameter. It should return 0 if there are no even digits.

For example, a program that uses the function *lastEven* follows.

```
int main() {
   cout << lastEven(23) << endl;       // prints 2
   cout << lastEven(1214) << endl;     // prints 4
   cout << lastEven(777) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int lastEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return x % 10;
   return lastEven(x / 10);
}
```

Problem 43 Write a function called *sumEven* that returns the sum of the even digits in a positive integer parameter.

For example, a program that uses the function *sumEven* follows.

```
int main() {
   cout << sumEven(23) << endl;       // prints 2
   cout << sumEven(1212) << endl;     // prints 4
   cout << sumEven(777) << endl;      // prints 0, because there are none
   return 0;
}
```

Answer:

```
int sumEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return x%10 + sumEven(x/10);
   return sumEven(x/10);
}
```

Problem 44 Write a function called *lastOdd* that returns the last odd digit in a positive integer parameter. It should return 0 if there are no odd digits.

For example, a program that uses the function *lastOdd* follows.

```
int main() {
   cout << lastOdd(23) << endl;       // prints 3
   cout << lastOdd(1254) << endl;     // prints 5
   cout << lastOdd(666) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int lastOdd(int x) {
   if (x <= 0) return 0;
   if (x % 2 != 0) return x%10;
   return lastOdd(x/10);
}
```

Problem 45 Write a function called *firstEven* that returns the first even digit in a positive integer parameter. It should return -1 if there are no even digits.

For example, a program that uses the function *firstEven* follows.

```
int main() {
   cout << firstEven(23) << endl;       // prints 2
   cout << firstEven(1416) << endl;     // prints 4
   cout << firstEven(777) << endl;      // prints -1
   return 0;
}
```

Answer:

```
int firstEven(int x) {
   if (x <= 0) return -1;
   if (firstEven(x/10) >= 0) return firstEven(x/10);
   if (x % 2 == 0) return x % 10;
   return -1;
}
```

Problem 46 Write a function called *evenLessOdd* that returns the sum of the even valued digit minus the sum of the odd valued digits in a positive integer parameter.

For example, a program that uses the function *evenLessOdd* follows.

```
int main() {
   cout << evenLessOdd(43) << endl;       // prints 1
   cout << evenLessOdd(9876) << endl;     // prints -2
   cout << evenLessOdd(777) << endl;      // prints -21
   return 0;
}
```

Answer:

```
int evenLessOdd(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 0) return evenLessOdd(x/10) + x % 10;
   return evenLessOdd(x/10) - x % 10;
}
```

Problem 47 Write a function called *firstOdd* that returns the first odd digit in a positive integer parameter. It should return -1 if there are no odd digits.

For example, a program that uses the function *firstOdd* follows.

```
int main() {
   cout << firstOdd(21) << endl;       // prints 1
   cout << firstOdd(3456) << endl;     // prints 3
   cout << firstOdd(666) << endl;      // prints -1
   return 0;
}
```

Answer:

```
int firstOdd(int x) {
   if (x <= 0) return -1;
   if (firstOdd(x/10) >= 0) return firstOdd(x/10);
   if (x % 2 == 1) return x % 10;
   return -1;
}
```

Problem 48 Write a function called *oddLessEven* that returns the sum of the odd valued digits minus the sum of the even valued digits in a positive integer parameter.

For example, a program that uses the function *oddLessEven* follows.

```
int main() {
   cout << oddLessEven(23) << endl;       // prints 1
   cout << oddLessEven(1234) << endl;     // prints -2
   cout << oddLessEven(777) << endl;      // prints 21
   return 0;
}
```

Answer:

```
int oddLessEven(int x) {
   if (x <= 0) return 0;
   if (x % 2 == 1) return x % 10 + oddLessEven(x/10);
   return oddLessEven(x/10) - x % 10;
}
```

Problem 49 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[2][3] = {{1,2,3}, {3,4,5}};
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;      // line (b)
    cout << x[x[0][0]][x[0][1]] << endl;                      // line (c)
    up(x,1,1);                                                // line (d)
    cout << up(x,1,2) << endl;                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
4
```

(b) What is the output at line (b)?

Answer:

```
1
4
```

(c) What is the output at line (c)?

Answer:

```
5
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
6
```

Problem 50 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (y < 2) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[2][3] = {{3,2,1}, {0,3,6}}, a = 0;
    cout << x[a][a] << endl;                                    // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][2 - i] << endl;    // line (b)
    cout << x[x[x[0][2]][0]][0] << endl;                        // line (c)
    up(x,1,1);                                                  // line (d)
    cout << up(x,1,2) << endl;                                  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
3
```

(b) What is the output at line (b)?

Answer:

```
1
3
```

(c) What is the output at line (c)?

Answer:

```
3
```

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

```
6
6
```

Problem 51 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (a[x][y] % 2 == 1) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[2][3] = {{0,1,2}, {4,5,6}}, a = 0;
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;      // line (b)
    cout << x[x[0][0]][x[0][1]] << endl;                      // line (c)
    cout << up(x,1,1) << endl;                                // line (d)
    up(x,1,2);                                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
5
```

(b) What is the output at line (b)?

Answer:

```
0
5
```

(c) What is the output at line (c)?

Answer:

```
1
```

(d) What is the output at line (d)?

Answer:

```
5
6
```

(e) What is the output at line (e)?

Answer:

Problem 52 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (y < 2) return a[1-x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[2][3] = {{2,1,0}, {0,4,8}}, a = 0;
    cout << x[a][2*a] << endl;                                    // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;          // line (b)
    cout << x[0][x[x[0][1]][0]]<< endl;                           // line (c)
    up(x,1,2);                                                    // line (d)
    cout << up(x,1,1) << endl;                                    // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
2
```

(b) What is the output at line (b)?

Answer:

```
2
4
```

(c) What is the output at line (c)?

Answer:

```
2
```

(d) What is the output at line (d)?

Answer:

```
8
```

(e) What is the output at line (e)?

Answer:

```
0
```

Problem 53 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}
```

```
int main() {
    int x[3][2] = {{1,2}, {3,3}, {4,5}};
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;   // line (b)
    cout << x[x[0][1]][x[0][0]] << endl;                      // line (c)
    up(x,1,1);                                                // line (d)
    cout << up(x,2,1) << endl;                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
3
```

(b) What is the output at line (b)?

Answer:

```
1
3
```

(c) What is the output at line (c)?

Answer:

```
5
```

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

```
6
```

Problem 54 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (y < 1) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[3][2] = {{3,2},{4,5},{0,1}}, a = 0;
    cout << x[a][a] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[2 - i][i] << endl;   // line (b)
    cout << x[x[x[2][0]][0]][0] << endl;                      // line (c)
    up(x,1,1);                                                // line (d)
    cout << up(x,2,1) << endl;                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
3
```

(b) What is the output at line (b)?

Answer:

```
0
5
```

(c) What is the output at line (c)?

Answer:

```
No output:
Code error, sorry
```

(d) What is the output at line (d)?

Answer:

```
5
```

(e) What is the output at line (e)?

Answer:

```
1
1
```

Problem 55 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[3][2] = {{0,1}, {3,4}, {5,7}};
    cout << x[1][1] << endl;                                  // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;   // line (b)
    cout << x[x[0][1]][x[0][0]] << endl;                      // line (c)
    up(x,1,1);                                                // line (d)
    cout << up(x,2,1) << endl;                                // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
4
```

(b) What is the output at line (b)?

Answer:

```
0
4
```

(c) What is the output at line (c)?

Answer:

```
3
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
8
```

Problem 56 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (y < 1) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[3][2] = {{2,3},{0,4},{1,5}}, a = 0;
    cout << x[a][a] << endl;                                    // line (a)
    for (int i = 0; i < 2; i++) cout << x[2 - i][i] << endl;    // line (b)
    cout << x[x[x[2][0]][0]][0] << endl;                        // line (c)
    up(x,1,1);                                                  // line (d)
    cout << up(x,2,1) << endl;                                  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
2
```

(b) What is the output at line (b)?

Answer:

```
1
4
```

(c) What is the output at line (c)?

Answer:

```
2
```

(d) What is the output at line (d)?

Answer:

```
4
```

(e) What is the output at line (e)?

Answer:

```
5
5
```

Problem 57 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   cout << sqrt("FFrreedd") << endl;      // prints: Fred
   cout << firstLetter("Freddy") << endl; // prints: F
   sort(a, 4);                            // prints: 159 265 314 358
   oddElements(a, 4);                     // prints: odd: 159 265
   a[0] = sum(a[1], a[2]);                // adds elements
   return 0;
}
```

(a) Title line for **sqrt**.

Answer:

```
string sqrt(string x)
```

(b) Title line for **firstLetter**.

Answer:

```
char firstLetter(string x)
```

(c) Title line for **sort**.

Answer:

```
void sort(int a[], int capacity)
```

(d) Title line for **oddElements**.

Answer:

```
void oddElements(int a[], int capacity)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 58 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
```

```
}

int main() {
  int x = 5, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
5
```

(b) line b:

```
15
```

(c) line c:

```
71
```

(d) line d:

```
37
```

(e) line e:

```
93
27
```

Problem 59 Write a function called *addThrees* that inserts a 3 after each digit of a positive integer parameter. For example, a program that uses the function *addThrees* follows.

```
int main() {
   cout << addThrees(3)  << endl;       // prints 33
   cout << addThrees(1313) << endl;     // prints 13331333
   cout << addThrees(777) << endl;      // prints 737373
   return 0;
}
```

Answer:

```
int addThrees(int x) {
   if (x == 0) return 0;
   return 100 * addThrees(x / 10) + 10 * (x % 10) + 3;
}
```

Problem 60 Write a C++ function called $halfs$ that divides each element of a 2-dimensional array (with two columns) by 2.
It should be possible to use your function in the following program.

```
main() {
   double data[2][2] = {{1, 2}, {3, 4}};
   halfs (data, 2, 2);
   for (int i = 0; i < 2; i++)
     cout << data[1][i] << " ";   // prints 1.5 2.0
}
```

Answer:

```
void halfs(double d[][2], int r, int c) {
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         d[i][j] = d[i][j] / 2;
      }
}
```

Problem 61 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   sqrt("FFrreedd");                    // prints: Fred
   firstLetter("Freddy");               // prints: F
   sort(a, 4);                          // prints: 159 265 314 358
   cout << oddElements(a, 4);           // prints: odd: 159 265
   swap(a[1], a[2]);                    // swaps elements
   return 0;
}
```

(a) Title line for **sqrt**.

Answer:

```
void sqrt(string x)
```

(b) Title line for **firstLetter**.

Answer:

```
void firstLetter(string x)
```

(c) Title line for **sort**.

Answer:

```
void sort(int a[], int capacity)
```

(d) Title line for **oddElements**.

Answer:

```
string oddElements(int a[], int capacity)
```

(e) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

Problem 62 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 4, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
4
```

(b) line b:

```
24
```

(c) line c:

```
62
```

(d) line d:

```
46
```

(e) line e:

```
84
32
```

Problem 63 Write a function called *addThrees* that inserts a 3 before each digit of a positive integer parameter. For example, a program that uses the function *addThrees* follows.

```
int main() {
   cout << addThrees(3)  << endl;       // prints 33
   cout << addThrees(1313) << endl;     // prints 31333133
   cout << addThrees(777) << endl;      // prints 373737
   return 0;
}
```

Answer:

```
int addThrees(int x) {
   if (x == 0) return 0;
   return 100 * addThrees(x / 10) + 30 + x % 10;
}
```

Problem 64 Write a C++ function called *roots* that replaces each element of an array by its root. It should be possible to use your function in the following program.

```
main() {
   double data[3] = {1.0, 4.0, 9.0};
   roots (data, 3);
   for (int i = 0; i < 3; i++)
     cout << data[i] << " ";   // prints 1 2 3
}
```

Answer:

```
#include <cmath>

void roots(double d[], int cap) {
   for (int i = 0; i < cap; i++)
      d[i] = sqrt(d[i]);
}
```

Problem 65 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   cout << firstLetter("Freddy") << endl; // prints: F
   cout << sqrt("FFrreedd") << endl;      // prints: Fred
   oddElements(a, 4);                     // prints: odd: 159 265
   sort(a, 4);                            // prints: 159 265 314 358
   a[0] = sum(a[1], a[2]);                // adds elements
   return 0;
}
```

(a) Title line for **firstLetter**.

Answer:

```
char firstLetter(string x)
```

(b) Title line for **sqrt**.

Answer:

```
string sqrt(string x)
```

(c) Title line for **oddElements**.

Answer:

```
void oddElements(int a[], int capacity)
```

(d) Title line for **sort**.

Answer:

```
void sort(int a[], int capacity)
```

(e) Title line for **sum**.

Answer:

```
int sum(int x, int y)
```

Problem 66 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 3, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
3
```

(b) line b:

```
13
```

(c) line c:

```
51
```

(d) line d:

```
35
```

(e) line e:

```
73
21
```

Problem 67 Write a function called *addTwos* that inserts a 2 after each digit of a positive integer parameter. For example, a program that uses the function *addTwos* follows.

```
int main() {
   cout << addTwos(3)  << endl;       // prints 32
   cout << addTwos(1212) << endl;     // prints 12221222
   cout << addTwos(777) << endl;      // prints 727272
   return 0;
}
```

Answer:

```
int addTwos(int x) {
   if (x == 0) return 0;
   return 100 * addTwos(x / 10) + 10 * (x % 10) + 2;
}
```

Problem 68 Write a C++ function called *squares* that replaces each element of a 2-dimensional array (with two columns) by its square.

It should be possible to use your function in the following program.

```
main() {
   int data[2][2] = {{1, 2}, {3, 4}};
   squares (data, 2, 2);
   for (int i = 0; i < 2; i++)
     cout << data[1][i] << " ";   // prints 9 16
}
```

Answer:

```
void squares(int d[][2], int r, int c) {
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         d[i][j] = d[i][j] * d[i][j];
      }
}
```

Problem 69 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   firstLetter("Freddy");                 // prints: F
   sqrt("FFrreedd");                      // prints: Fred
   cout << oddElements(a, 4);             // prints: odd: 159 265
   sort(a, 4);                            // prints: 159 265 314 358
   swap(a[1], a[2]);                      // swaps elements
   return 0;
}
```

(a) Title line for **firstLetter**.

Answer:

```
void firstLetter(string x)
```

(b) Title line for **sqrt**.

Answer:

```
void sqrt(string x)
```

(c) Title line for **oddElements**.

Answer:

```
string oddElements(int a[], int capacity)
```

(d) Title line for **sort**.

Answer:

```
void sort(int a[], int capacity)
```

(e) Title line for **swap**.

Answer:

```
void swap(int &x, int &y)
```

Problem 70 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
2
```

(b) line b:

```
22
```

(c) line c:

```
42
```

(d) line d:

```
44
```

(e) line e:

```
64
24
```

Problem 71 Write a function called *addTwos* that inserts a 2 before each digit of a positive integer parameter. For example, a program that uses the function *addTwos* follows.

```
int main() {
   cout << addTwos(3)  << endl;       // prints 23
   cout << addTwos(1212) << endl;     // prints 21222122
   cout << addTwos(777) << endl;      // prints 272727
   return 0;
}
```

Answer:

```
int addTwos(int x) {
   if (x == 0) return 0;
   return 100 * addTwos(x / 10) + 20 + x % 10;
}
```

Problem 72 Write a C++ function called *cubes* that replaces each element of an array by its cube. It should be possible to use your function in the following program.

```
main() {
   int data[3] = {1, 2, 3};
   cubes (data, 3);
   for (int i = 0; i < 3; i++)
     cout << data[i] << " ";   // prints 1 8 27
}
```

Answer:

```
void cubes(int d[], int cap) {
   for (int i = 0; i < cap; i++)
      d[i] = d[i] * d[i] * d[i];
}
```

Problem 73 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   undouble(11223344);                    // prints: 1234
   firstDigit(65536);                     // prints: Six
   printSorted(a, 4);                     // prints: Freddy Jack Kelly Max
   cout << join(a[1], a[3]) << endl;      // prints: MaxJack
   randomWords(a, 4);                     // assigns new random values to array
   return 0;
}
```

(a) Title line for **undouble**.

Answer:

```
void undouble(int x)
```

(b) Title line for **firstDigit**.

Answer:

```
void firstDigit(int x)
```

(c) Title line for **printSorted**.

Answer:

```
void printSorted(string a[], int capacity)
```

(d) Title line for **join**.
Answer:

```
string join(string x, string y)
```

(e) Title line for **randomWords**.
Answer:

```
void randomWords(string a[], int capacity)
```

Problem 74 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 5, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
5
```

(b) line b:

```
62
```

(c) line c:

```
02
```

(d) line d:

```
17
```

(e) line e:

```
73
21
```

Problem 75 Write a function called *killTwos* that deletes all digits that are multiples of 2 from a positive integer parameter.

For example, a program that uses the function *killTwos* follows.

```
int main() {
   cout << killTwos(11) << endl;       // prints 11
   cout << killTwos(1212) << endl;     // prints 11
   cout << killTwos(2400) << endl;     // prints 0, because no digits are left
   return 0;
}
```

Answer:

```
int killTwos(int x) {
   if (x == 0) return 0;
   if ((x % 10) % 2 == 0) return killTwos(x / 10);
   return 10 * killTwos(x / 10) + x % 10;
}
```

Problem 76 Write a C++ function called *numOdd* that returns the number of odd elements in a 2-dimensional array (with 4 columns).

It should be possible to use your function in the following program. (The output from this program is 2 because only the two 11s are odd).

```
main() {
   int data[2][4] = {{11, 12, 14, 0}, {32, 12, 132, 11}};
   int x;
   x = numOdd (data, 2, 4);
     //   data is the 2-d array, 2 and 4 are its capacities
   cout << "The number of odds is: " << x  << endl;
}
```

Answer:

```
int numOdd(int d[][4], int r, int c) {
   int count = 0;
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         if ((d[i][j] % 2) != 0) count++;
      }
   return count;
}
```

Problem 77 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   cout << undouble(11223344);            // prints: 1234
   cout << firstDigit(65536) << endl;     // prints: Six
   sort(a, 4);                            // prints: Freddy Jack Kelly Max
   cout << halfString(a[0]) << endl;      // prints: Fre
   a[1] = randomWord();                   // assigns a random value
   return 0;
}
```

(a) Title line for **undouble**.
Answer:

```
int undouble(int x)
```

(b) Title line for **firstDigit**.
Answer:

```
string firstDigit(int x)
```

(c) Title line for **sort**.
Answer:

```
void sort(string a[], int capacity)
```

(d) Title line for **halfString**.
Answer:

```
string halfString(string x)
```

(e) Title line for **randomWord**.
Answer:

```
string randomWord()
```

Problem 78 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 4, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
4
```

(b) line b:

(c) line c:

```
12
```

(d) line d:

```
26
```

(e) line e:

```
63
18
```

Problem 79 Write a function called *twos* that deletes all digits that are not multiples of 2 from a positive integer parameter.

For example, a program that uses the function *twos* follows.

```
int main() {
   cout << twos(23) << endl;       // prints 2
   cout << twos(1212) << endl;     // prints 22
   cout << twos(777) << endl;      // prints 0, because nothing is left
   return 0;
}
```

Answer:

```
int twos(int x) {
   if (x == 0) return 0;
   if ((x % 10) % 2 != 0) return twos(x / 10);
   return 10*twos(x / 10) + x % 10;
}
```

Problem 80 Write a C++ function called *range* that returns the difference between the largest and smallest elements in a 2-dimensional array (with 4 columns).

It should be possible to use your function in the following program. (The output from this program is 10 because the difference between the largest element 13 and the smallest element 3 is $13 - 3 = 10$).

```
main() {
   int data[2][4] = {{11, 12, 11, 5}, {6, 3, 12, 13}};
   int x;
   x = range (data, 2, 4);
     //   data is the 2-d array, 2 and 4 are its capacities
   cout << "The range is: " << x  << endl;
}
```

Answer:

```
int range(int d[][4], int r, int c) {
   int max = d[0][0];
   int min = d[0][0];
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
          if (d[i][j] < min) min = d[i][j];
          if (d[i][j] > max) max = d[i][j];
      }
   return max - min;
}
```

Problem 81 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   firstDigit(65536);                    // prints: Six
   undouble(11223344);                   // prints: 1234
   cout << join(a[1], a[3]) << endl;     // prints: MaxJack
   printSorted(a, 4);                    // prints: Freddy Jack Kelly Max
   randomWords(a, 4);                    // assigns new random values to array
   return 0;
}
```

(a) Title line for **firstDigit**.

Answer:

```
void firstDigit(int x)
```

(b) Title line for **undouble**.

Answer:

```
void undouble(int x);
```

(c) Title line for **join**.

Answer:

```
string join(string x, string y)
```

(d) Title line for **printSorted**.

Answer:

```
void printSorted(string a[], int capacity)
```

(e) Title line for **randomWords**.

Answer:

```
void randomWords(string a[], int capacity)
```

Problem 82 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 3, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
3
```

(b) line b:

```
42
```

(c) line c:

```
02
```

(d) line d:

```
15
```

(e) line e:

```
53
15
```

Problem 83 Write a function called *killTwos* that deletes all digits that are equal to 2 from a positive integer parameter.

For example, a program that uses the function *killTwos* follows.

```
int main() {
   cout << killTwos(11) << endl;       // prints 11
   cout << killTwos(1212) << endl;     // prints 11
   cout << killTwos(222) << endl;      // prints 0, because no digits are left
   return 0;
}
```

Answer:

```
int killTwos(int x) {
   if (x == 0) return 0;
   if (x % 10 == 2) return killTwos(x / 10);
   return 10 * killTwos(x / 10) + x % 10;
}
```

Problem 84 Write a C++ function called *numEven* that returns the number of even elements in a 2-dimensional array (with 3 columns).

It should be possible to use your function in the following program. (The output from this program is 2 because only the two 12s are even).

```
main() {
   int data[2][3] = {{11, 12, 11}, {3, 12, 13}};
   int x;
   x = numEven (data, 2, 3);
     //   data is the 2-d array, 2 and 3 are its capacities
   cout << "The number of evens is: " << x  << endl;
}
```

Answer:

```
int numEven(int d[][3], int r, int c) {
   int count = 0;
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
         if ((d[i][j] % 2) == 0) count++;
      }
   return count;
}
```

Problem 85 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   cout << firstDigit(65536) << endl;     // prints: Six
   cout << undouble(11223344);            // prints: 1234
   cout << halfString(a[0]) << endl;      // prints: Fre
   sort(a, 4);                            // prints: Freddy Jack Kelly Max
   a[1] = randomWord();                   // assigns a random value
   return 0;
}
```

(a) Title line for **firstDigit**.

Answer:

```
string firstDigit(int x)
```

(b) Title line for **undouble**.

Answer:

```
int undouble(int x)
```

(c) Title line for **halfString**.

Answer:

```
string halfString(string x)
```

(d) Title line for **sort**.

Answer:

```
void sort(string a[], int capacity)
```

(e) Title line for **randomWord**.

Answer:

```
string randomWord()
```

Problem 86 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
```

```
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
2
```

(b) line b:

```
32
```

(c) line c:

```
12
```

(d) line d:

```
24
```

(e) line e:

```
43
12
```

Problem 87 Write a function called *twos* that deletes all digits that are not equal to 2 from a positive integer parameter.

For example, a program that uses the function *twos* follows.

```
int main() {
   cout << twos(23) << endl;       // prints 2
   cout << twos(1212) << endl;     // prints 22
   cout << twos(777) << endl;      // prints 0, because nothing is left
   return 0;
}
```

Answer:

```
int twos(int x) {
   if (x == 0) return 0;
   if (x % 10 != 2) return twos(x / 10);
   return 10*twos(x / 10) + 2;
}
```

Problem 88 Write a C++ function called *range* that returns the difference between the largest and smallest elements in a 2-dimensional array (with 3 columns).

It should be possible to use your function in the following program. (The output from this program is 10 because the difference between the largest element 13 and the smallest element 3 is $13 - 3 = 10$).

```
main() {
   int data[2][3] = {{11, 12, 11}, {3, 12, 13}};
   int x;
   x = range (data, 2, 3);
     //   data is the 2-d array, 2 and 3 are its capacities
   cout << "The range is: " << x  << endl;
}
```

Answer:

```
int range(int d[][3], int r, int c) {
   int max = d[0][0];
   int min = d[0][0];
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++) {
          if (d[i][j] < min) min = d[i][j];
          if (d[i][j] > max) max = d[i][j];
      }
   return max - min;
}
```

Problem 89 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {3,1,4,1,5};
   int x[2][3] = {{0,1,3},{2,4,5}};
   string s= "Hello";
   string t;

   cout << average(a, 5) << endl;           // prints the average:  2.8
   t = reverse(s);  cout << t << endl;      // prints:  olleH
   reverseRows(x, 2, 3);                    // prints:  2 4 5, 0 1 3
   if (hasRepeat(a, 5)) cout << "Has repeat" << endl;
                                            // prints:  Has repeat
   t = entries(a, 5); cout << t << endl;    // prints:  3,1,4,1,5
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int a[], int cap)
```

(b) Title line for **reverse**

Answer:

```
string reverse(string s)
```

(c) Title line for **reverseRows**

Answer:

```
void reverseRows(int x[][3], int r, int c)
```

(d) Title line for **hasRepeat**

Answer:

```
bool hasRepeat(int a[], int cap)
```

(e) Title line for **entries**

Answer:

```
string entries(int a[], int cap)
```

Problem 90 Consider the following C++ program.

```
#include <iostream>
using namespace std;

char f(string s, int n) {
   if (n >= s.length()) return 'A';
   return s[n];
}

int mystery (int x) {
   if (x > 5) return 0;
   cout << -x;
   return x;
}

int main () {
   cout << f("Hello", 20) << endl;     //line A
   cout << f("Hello", 1)  << endl;     //line B
   cout << mystery(19683) << endl;     //line C
   cout << mystery(2) << endl;         //line D
   mystery(-5);                        //line E
   cout << endl;
   return 0;
}
```

(a) What is the output at line A?

Answer:

```
A
```

(b) What is the output at line B?

Answer:

```
e
```

(c) What is the output at line C?

Answer:

```
0
```

(d) What is the output at line D?

Answer:

```
-22
```

(e) What is the output at line E?

Answer:

```
5
```

Problem 91 Write a function called *extraOne* that places an initial 1 at the start of an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *extraOne* follows.

```
int main() {
   int x = extraOne(729);
   cout << x << endl;    // prints   1729
   return 0;
}
```

Answer:

```
int extraOne(int x) {
   if (x < 10) return 10 + x;
   return 10 * extraOne(x / 10) + x % 10;
}
```

Problem 92 Write a function called *dropDimension* that copies the entries from a 2-dimensional array row by row as the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[100];
   int y[2][3] = {{3,1,4}, {1,5,9}};
   int yrows = 2, ycols = 3;
   dropDimension(y, yrows, ycols, x);
   for (int i = 0; i <= 5; i++) cout << x[i];
     // 314159   is printed
   cout << endl;
   return 0;
}
```

Answer:

```
void dropDimension(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int r = 0; r < rows; r++)
      for (int c = 0; c < cols; c++) {
          x[i] = y[r][c];
          i++;
      }
}
```

Problem 93 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {3,1,4,1,5};
   int x[2][3] = {{0,1,3},{2,4,8}};
   string s= "Hello";
   string t;

   cout << average(x, 2, 3) << endl;        // prints the average:  3.0
   t = doubleIt(s);  cout << t << endl;     // prints:  HelloHello
   reverseCols(x, 2, 3);                    // prints:  3 0 1, 8 4 2
   if (isPositive(a[0])) cout << "Positive" << endl;
                                            // prints:  Positive
   cout << midEntry(a, 5) << endl;          // prints:  4
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int x[][3], int r, int c)
```

(b) Title line for **doubleIt**

Answer:

```
string doubleIt(string s)
```

(c) Title line for **reverseCols**

Answer:

```
void reverseCols(int x[][3], int r, int c)
```

(d) Title line for **isPositive**

Answer:

```
bool isPositive(int x)
```

(e) Title line for **midEntry**

Answer:

```
int midEntry(int a[], int cap)
```

Problem 94 Consider the following C++ program.

```
#include <iostream>
using namespace std;

string f(string s, int n) {
   if (n >= s.length()) return "XYZ";
   return s.substr(n);
}

int mystery (int x) {
   if (x > 5) return 0;
   return x;
}

int main () {
   cout << mystery(19683) << endl;     //line A
```

```
   cout << mystery(2) << endl;         //line B
   cout << f("Hello", 20) << endl;     //line C
   cout << f("Hello", 1)  << endl;     //line D
   mystery(-5);                        //line E
   return 0;
}
```

(a) What is the output at line A?

Answer:

```
0
```

(b) What is the output at line B?

Answer:

```
2
```

(c) What is the output at line C?

Answer:

```
XYZ
```

(d) What is the output at line D?

Answer:

```
ello
```

(e) What is the output at line E?

Answer:

Problem 95 Write a function called *doubleEight* that places an extra digit 8 after the last 8 in an integer parameter. If there is no 8 present, nothing is done. (Assume that the input parameter is not negative.)

For example, a program that uses the function *doubleEight* follows.

```
int main() {
   int x = doubleEight(19683);
   cout << x << endl;                      // prints   196883
   cout << doubleEight(271828) << endl;    // prints   2718288
   cout << doubleEight(314159) << endl;    // prints   314159
   return 0;
}
```

Answer:

```
int doubleEight(int x) {
   if (x % 10 == 8) return 10 * x + 8;
   if (x < 10) return x;
   return 10 * doubleEight(x / 10) + x % 10;
}
```

Problem 96 Write a function called *dropDimension* that copies the entries from a 2-dimensional array column by column as the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[100];
   int y[2][3] = {{3,4,5}, {1,1,9}};
   int yrows = 2, ycols = 3;
   dropDimension(y, yrows, ycols, x);
   for (int i = 0; i <= 5; i++) cout << x[i];
     // 314159   is printed
   cout << endl;
   return 0;
}
```

Answer:

```
void dropDimension(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int c = 0; c < cols; c++)
      for (int r = 0; r < rows; r++) {
          x[i] = y[r][c];
          i++;
      }
}
```

Problem 97 Write a function called *extraTwo* that inserts an extra digit 2 as the second digit of an integer parameter. (Assume that the input parameter is positive.)

For example, a program that uses the function *extraTwo* follows.

```
int main() {
   int x = extraTwo(79);
   cout << x << endl;              // prints   729
   cout << extraTwo(1) << endl;    // prints   12
   return 0;
}
```

Answer:

```
int extraTwo(int x) {
   if (x < 10) return 10 * x + 2;
   return 10 * extraTwo(x / 10) + x % 10;
}
```

Problem 98 Write a function called $fill2D$ that fills the entries of a 2-dimensional array column by column from the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[11] = {3,1,4,1,5,9,2,6,5,3,5};
   int y[2][3];
   int yrows = 2, ycols = 3;
   fill2D(y, yrows, ycols, x);
   for (int i = 0; i < yrows; i++) {
      for (int j = 0; j < ycols; j++) cout << y[i][j];
      cout << endl;
   }
     // 345   is printed
     // 119
   return 0;
}
```

Answer:

```
void fill2D(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int c = 0; c < cols; c++)
      for (int r = 0; r < rows; r++) {
          y[r][c] = x[i];
          i++;
      }
}
```

Problem 99 Write a function called *doubleFour* that places an extra copy of the 4th digit right after that digit in an integer parameter. If there is no 4th digit, nothing is done. (Assume that the input parameter is not negative.)

For example, a program that uses the function *doubleFour* follows.

```
int main() {
   int x = doubleFour(19683);
   cout << x << endl;              // prints    196883
   cout << doubleFour(271828);     // prints   2718828
   cout << doubleFour(314159);     // prints   3141159
   return 0;
}
```

Answer:

```
int doubleFour(int x) {
   if (x < 1000) return x;
   if (x < 10000) return 10 * x + x % 10;
   return 10 * doubleFour(x / 10) + x % 10;
}
```

Problem 100 Write a function called $fill2D$ that fills the entries of a 2-dimensional array row by row from the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[11] = {3,1,4,1,5,9,2,6,5,3,5};
   int y[2][3];
   int yrows = 2, ycols = 3;
   fill2D(y, yrows, ycols, x);
   for (int i = 0; i < yrows; i++) {
      for (int j = 0; j < ycols; j++) cout << y[i][j];
      cout << endl;
   }
     // 314   is printed
     // 159
   return 0;
}
```

Answer: Answer:

```
void fill2D(int y[][3], int rows, int cols, int x[]) {
   int i = 0;
   for (int r = 0; r < rows; r++)
```

```
      for (int c = 0; c < cols; c++) {
          y[r][c] = x[i];
          i++;
      }
}
```

Problem 101 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int i = 3, j = 5;
   int a[9] = {3,1,4,1,5,9,2,6,5};
   int x[3][2] = {{0,1},{3,2},{4,5}};

   cout << min(i, j) << endl;              // prints minimum
   printArray(x, 3, 2);                    // prints array
   cout << average(a, 9) << endl;          // prints average
   swap(a, 3, 5);                          // swap elements 3 and 5
   reverse(a[1]);                          // reverse the digits in a[1]
   return 0;
}
```

(a) Title line for **min**

Answer:

```
int min (int i, int j) {
```

(b) Title line for **printArray**

Answer:

```
void printArray(int a[][2], int rows, int cols)
```

(c) Title line for **average**

Answer:

```
double average(int a[], int cap)
```

(d) Title line for **swap**

Answer:

```
void swap(int a[], int i, int j )
```

(e) Title line for **reverse**

Answer:

```
void reverse(int &a)
```

Problem 102 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive (int n) {
   if (n < 10) return n;
   return 100 * recursive (n / 100) + 10 * (n % 10);
```

```
}

int mystery (int x) {
   cout << x << "54321";
   return x;
}

int main () {
   cout << recursive (7) << endl;     //line A
   cout << recursive (135) << endl;   //line B
   cout << recursive (19683) << endl; //line C
   cout << mystery (2) << endl;       //line D
   mystery (2);                       //line E
   return 0;
}
```

(a) What is the output at line A?

Answer:

7

(b) What is the output at line B?

Answer:

150

(c) What is the output at line C?

Answer:

16030

(d) What is the output at line D?

Answer:

2543212

(e) What is the output at line E?

Answer:

254321

Problem 103 Write a function called *smallestDigit* that finds the smallest digit in an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *smallestDigit* follows.

```
int main() {
   cout << smallestDigit(29) << endl;    // prints    2
   cout << smallestDigit(31415) << endl; // prints    1
   cout << smallestDigit(7) << endl;     // prints    7
   return 0;
}
```

Answer:

```
int smallestDigit(int x) {
   if (x < 10) return x;
   int ans = smallestDigit(x/10);
   if (ans < x % 10) return ans;
   return x % 10;
}
```

Problem 104 Write a function called *lastIndex* that finds the largest index of an entry in an array of integers that matches a given target. If the target is not present the function should return an answer of -1.

For example, a program that uses the function follows.

```
int main() {
   int x[6] = {3, 1, 4, 1, 5, 9};
   int capacity = 6;
   int target = 5;
   cout << lastIndex(x, capacity, target) << endl;
     // prints  4  because the target 5 is found as element number 4
   cout << lastIndex(x, capacity, 1) << endl;
     // prints  3  because the target 1 is last found as element number 3
   cout << lastIndex(x, capacity, 8) << endl;
     // prints  -1  because the target 8 is not found.
   return 0;
}
```

Answer:

```
int lastIndex(int a[], int capacity, int target) {
   for (int i = capacity - 1; i >= 0; i--)
      if (a[i] == target) return i;
   return -1;
}
```

Problem 105 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int i = 3, j = 5;
   int a[9] = {3,1,4,1,5,9,2,6,5};
   int x[3][2] = {{0,1},{3,2},{4,5}};

   cout << average(i, j) << endl;          // prints average
   printArray(a, 9);                       // prints array
   cout << min(x, 3, 2) << endl;           // prints minimal element
   reverse(a, 9);                          // reverse the order of elements
   swap(a[1], a[2]);                       // swap two values
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int i, int j)
```

(b) Title line for **printArray**

Answer:

```
void printArray(int a[], int cap)
```

(c) Title line for **min**

Answer: int min(int a[][2], int rows, int cols)

(d) Title line for **reverse**
Answer:

```
void reverse(int a[], int cap)
```

(e) Title line for **swap**
Answer:

```
void swap(int &i, int &j)
```

Problem 106 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive (int n) {
   if (n < 10) return n;
   return 100 * recursive (n / 100) + 11 * (n % 10);
}

int mystery (int x) {
   cout << x << "12345";
   return x;
}

int main () {
   cout << recursive (7) << endl;     //line A
   cout << recursive (135) << endl;   //line B
   cout << recursive (19683) << endl; //line C
   cout << mystery (2) << endl;       //line D
   mystery (2);                       //line E
   return 0;
}
```

(a) What is the output at line A?
Answer:

```
7
```

(b) What is the output at line B?
Answer:

```
155
```

(c) What is the output at line C?
Answer:

```
16633
```

(d) What is the output at line D?
Answer:

```
2123452
```

(e) What is the output at line E?
Answer:

```
212345
```

Problem 107 Write a function called *biggestDigit* that finds the biggest digit in an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *biggestDigit* follows.

```
int main() {
   cout << biggestDigit(29) << endl;    // prints    9
   cout << biggestDigit(31415) << endl; // prints    5
   cout << biggestDigit(7) << endl;     // prints    7
   return 0;
}
```

Answer:

```
int biggestDigit(int x) {
   if (x < 10) return x;
   int ans = biggestDigit(x/10);
   if (ans > x % 10) return ans;
   return x % 10;
}
```

Problem 108 Write a function called $firstIndex$ that finds the smallest index of an entry in an array of integers that matches a given target. If the target is not present the function should return an answer of -1.

For example, a program that uses the function follows.

```
int main() {
   int x[6] = {3, 1, 4, 1, 5, 9};
   int capacity = 6;
   int target = 5;
   cout << firstIndex(x, capacity, target) << endl;
     // prints  4  because the target 5 is found as element number 4
   cout << firstIndex(x, capacity, 1) << endl;
     // prints  1  because the target 1 is first found as element number 1
   cout << firstIndex(x, capacity, 8) << endl;
     // prints  -1  because the target 8 is not found.
   return 0;
}
```

Answer:

```
int firstIndex(int a[], int capacity, int target) {
   for (int i = 0; i < capacity; i++)
      if (a[i] == target) return i;
   return -1;
}
```

Problem 109 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int a[4] = {3,1,4,1}, i = 3, j = 5, k = 4;
   int b[4] = {2,7,1,8};
   int x[2][2] = {{0,1},{3,2}};

   cout << max(i, j, k) << endl;          // prints maximum
   printMax(a, 4);                        // prints maximum
```

```
   cout << max2d(x, 2, 2) << endl;        // prints maximum
   swap(i, j);                            // swap
   swapArrays(a, b, 4);                   // swap first 4 elements in arrays
   return 0;
}
```

(a) Title line for **max**

Answer:

```
int max(int a, int b, int c)
```

(b) Title line for **printMax**

Answer:

```
void printMax(int a[], int cap)
```

(c) Title line for **max2d**

Answer:

```
int max2d(int a[][2], int r, int c)
```

(d) Title line for **swap**

Answer:

```
void swap(int &a, int &b)
```

(e) Title line for **swapArrays**

Answer:

```
void swapArrays(int a[], int b[], int n)
```

Problem 110 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int main() {
  int x;
  cout << "Enter an integer:";
  cin >> x;

  if (x > 0) cout << "Goodbye" << endl;
  if (x < -10) {
     cout << x + 2 << endl;
     return 0;
  }
  else if (x % 2 != 0) cout << "odd" << endl;

  for (int i = 1; i < x; i++) cout << i;
  cout << endl;
  for (int i = 1; i <= -x; i++) {
       for (int j = 1; j <= 3; j++) cout << "*";
       cout << endl;
  }

  return 0;
}
```

(a) What is the output if the user enters *-729*?

Answer:

```
-727
```

(b) What is the output if the user enters *4*?

Answer:

```
Goodbye
123
```

(c) What is the output if the user enters *-5*?

Answer:

```
odd

***
***
***
***
***
```

(d) What is the output if the user enters *-4*?

Answer:

```
***
***
***
***
```

(e) What is the output if the user enters *3*?

Answer:

```
Goodbye
odd
12
```

Problem 111 Write a function called *doubleFirst* that places an extra copy of the first digit at the start of a number.

For example, a program that uses the function *doubleFirst* follows.

```
int main() {
   cout << doubleFirst(29) << endl;    // prints    229
   cout << doubleFirst(19683) << endl; // prints 119683
   cout << doubleFirst(9) << endl;     // prints     99
   return 0;
}
```

Answer:

```
int doubleFirst(int x) {
   if (x < 10) return 11*x;
   return doubleFirst(x / 10) * 10 + x % 10;
}
```

Problem 112 Write a function called *findLargest* that finds the largest possibility for the sum of the entries in a row of a 2-dimensional array of integers. The array and the capacities are parameters.

For example, a program that uses the function follows.

```
int main() {
   int d[2][3] = {{2,4,6}, {1,3,5}};
   cout << findLargest(d, 2, 3) << endl;
     // prints    12, because the sum 12 = 2+4+6 is larger than 1+3+5
   return 0;
}
```

Answer:

```
int findLargest(int d[][3], int r, int c) {
   int value = 0;
   for (int col = 0; col < c; col++)
      value = value + d[0][col];
   for (int row = 0; row < r; row++) {
      int rowValue = 0;
      for (int col = 0; col < c; col++)
         rowValue = rowValue + d[row][col];
      if (rowValue > value) value = rowValue;
   }
   return value;
}
```

Problem 113 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int a[4] = {3,1,4,1}, i = 3, j = 5, k = 4;
   int x[2][2] = {{0,1},{3,2}};

   cout << average(i, j, k) << endl;          // prints average
   printAverage(a, 4);                        // prints average
   cout << average2d(x, 2, 2) << endl;        // prints average
   sort(i, j ,k );                            // sort into order
   sort3(a, 4);                               // sort into order
   return 0;
}
```

(a) Title line for **average**

Answer:

```
double average(int a, int b, int c)
```

(b) Title line for **printAverage**

Answer:

```
void printAverage(int a[], int cap)
```

(c) Title line for **average2d**

Answer:

```
double average2d(int x[][2], int r, int c)
```

(d) Title line for **sort**

Answer:

```
void sort(int &a, int &b, int &c)
```

(e) Title line for **sort3**

Answer:

```
void sort3(int a[], int cap)
```

Problem 114 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int main() {
  int x;
  cout << "Enter an integer:";
  cin >> x;

  if (x < 0) cout << "Goodbye" << endl;
  if (x > 10) {
     cout << x % 10 << endl;
     return 0;
  }
  else if (x % 2 != 0) cout << "odd" << endl;

  for (int i = 1; i <= x; i++) cout << i;
  cout << endl;
  for (int i = 1; i < -x; i++) {
       for (int j = 1; j < 3; j++) cout << "*";
       cout << endl;
  }

  return 0;
}
```

(a) What is the output if the user enters *729*?

Answer:

```
9
```

(b) What is the output if the user enters *9*?

Answer:

```
odd
123456789
```

(c) What is the output if the user enters *5*?

Answer:

```
odd
12345
```

(d) What is the output if the user enters *4*?

Answer:

```
1234
```

(e) What is the output if the user enters *-3*?

Answer:

```
Goodbye
odd

**
**
```

Problem 115 Write a function called *dropSecond* that removes the second digit of an integer parameter. (Assume that the input parameter is not negative. If the parameter has just one digit, return that digit.)

For example, a program that uses the function *dropSecond* follows.

```
int main() {
   cout << dropSecond(29) << endl;    // prints    2, the 9 dropped
   cout << dropSecond(19683) << endl; // prints 1683, the 9 dropped
   cout << dropSecond(9) << endl;     // prints    9
   return 0;
}
```

Answer:

```
int dropSecond(int x) {
   if (x < 10) return x;
   if (x < 100) return x / 10;
   return dropSecond(x / 10) * 10 + x % 10;
}
```

Problem 116 Write a function called *findLargest* that finds the largest entry in a specified column of a 2-dimensional array of integers. The array, the capacities, and the specified column are parameters.

For example, a program that uses the function follows.

```
int main() {
   int d[2][3] = {{2,4,6}, {1,3,5}};
   cout << findLargest(d, 2, 3, 0) << endl;
     // prints    2, because this is the largest entry in column 0
   return 0;
}
```

Answer:

```
int findLargest(int d[][3], int r, int c, int x) {
   int ans = d[0][x];
   for (int row = 0; row < r; row++)
      if (d[row][x] > ans) ans = d[row][x];
   return ans;
}
```

Problem 117 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **num7s** which returns the number of digits equal to 7 in an input integer.

Answer:

```
int num7s(int x)
```

(b) A function called **num7s** which returns the number of elements equal to 7 in an input array of integers.

Answer:

```
int num7s(int x[], int capacity)
```

(c) A function called **num7s** which returns the number of characters equal to 7 in an input string.

Answer:

```
int num7s(string x)
```

(d) A function called **num7s** which changes an integer parameter to be the number of 7's in its decimal expansion. (For example if the input is 777111 it would be changed to 3 because it has 3 digits equal to 7.)

Answer:

```
void num7s(int &x)
```

(e) A function called **num7s** which returns the number of elements equal to 7 in a 2-dimensional array of integers with size 7×7.

Answer:

```
int num7s(int x[][7], int rows, int cols)
```

Problem 118 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x) {
   if (x <= 0) return 0;
   if (x >= 9 && x % 2 == 1) return x - 1;
   if (x >= 9 || x % 3 == 0) return x - 2;
   return 7;
}

int rec(int x) {
  if (x < 10) return fun(x);
  return rec(x / 10) + rec(x % 10);
}

int main() {
    cout << fun(3) << endl;    // line (a)
    cout << fun(30) << endl;   // line (b)
    cout << fun(33) << endl;   // line (c)
    cout << rec(33) << endl;   // line (d)
    cout << rec(999) << endl;  // line (e)
}
```

(a) What is the output at line (a)?

Answer:

1

(b) What is the output at line (b)?

Answer:

28

(c) What is the output at line (c)?

Answer:

32

(d) What is the output at line (d)?

Answer:

```
2
```

(e) What is the output at line (e)?

Answer:

```
24
```

Problem 119 Write a function called *startBinary* that returns a number giving the first 2 digits in the binary expansion of an integer parameter. (Assume that the input parameter is not negative. If the parameter has just one binary digit, return that digit.)

For example, a program that uses the function *startBinary* follows.

```
int main() {
   int x = startBinary(6);
   cout << x << endl;               // prints 11 because 6 in binary is 110
   cout << startBinary(23) << endl; // prints 10 because 23 is 10111 in binary
   cout << startBinary( 3) << endl; // prints 11 because  3 is    11 in binary
   cout << startBinary( 1) << endl; // prints  1 because  1 is     1 in binary
   return 0;
}
```

Answer:

```
int startBinary(int x) {
   if (x < 2) return x;
   if (x == 2) return 10;
   if (x == 3) return 11;
   return startBinary(x/2);
}
```

Problem 120 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter a positive integer n that is less than 100. If the user enters an incorrect value, the program terminates. The program next asks the user to enter n^2 strings to be stored in a 2-dimensional array with size $n \times n$. The program then reports the maximum number of times that it can find the string *Kamil* in any row or column of the array.

For example, if the user enters 4 for n and then enters the 16 strings:

```
Kamil Peter  Dustin Kamil
Kamil Andrew Carl   Phil
Rat   Rat    Rat    Rat
Kamil Peter  Dustin Kamil
```

The final output would be 3 because Kamil appears three times in the first column, and no more than three times in any row or column.

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int n;
```

```
   cout << "Enter a positive integer that is less than 100: ";
   cin >> n;
   if (n < 1 || n > 99) exit(1);

   string data[100][100];
   cout << "Enter " << n * n << " data items (each is a string)\n";
   for (int i = 0; i < n; i++)
     for (int j = 0; j < n; j++)
         cin >> data[i][j];

   int max = 0;
   for (int row = 0; row < n; row++) {
      int rowKamils = 0;
      for (int col = 0; col < n; col++)
          if (data[row][col] == "Kamil") rowKamils ++;
      if (rowKamils > max) max = rowKamils;
   }

   for (int col = 0; col < n; col++) {
      int colKamils = 0;
      for (int row = 0; row < n; row++)
          if (data[row][col] == "Kamil") colKamils ++;
      if (colKamils > max) max = colKamils;
   }

   cout << "The maximal number of Kamils in a row or column is " << max << endl;
   return 0;
}
```

Problem 121 Write header lines (prototypes) for the following functions. **Do not attempt to supply the blocks for the functions**.

(a) A function called **isNegative** that tests whether a decimal number is negative.

Answer:

```
bool isNegative(double x)
```

(b) A function called **thirdChar** which uses a string as input and returns the third character in the string.

Answer:

```
char thirdChar(string s)
```

(c) A function called **swapLast2** which modifies an array of integers. The task of the function is to swap the last two elements of the array.

Answer:

```
void swapLast2(int a[], int cap)
```

(d) A function called **printPic** which uses as input an 6×6 array of characters that represents a picture. The task of the function is to print the picture.

Answer:

```
void printPic(char pic[][6], int rows, int cols)
```

(e) A function called **reverseArray** which is to reverse the order of elements in an array of integers.

Answer:

```
void reverseArray(int x[], int cap)
```

Problem 122 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int data[], int p, int q) {
  data[p] = data[q];
  data[q] = data[p];
}

void m2(int &p, int q) {
  int temp = p;
  p = q;
  q = temp;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int x[8] = {0, 1, 2, 3, 4, 5, 6, 7};
  int y[7] = {0, 1, 2, 3, 4, 5, 6};
  int a = 3, b = 4;

  print(x, 3);                          // line (a)
  mystery(x, 1, 2);  print(x, 5);       // line (b)
  for (int i = 1; i <= 7; i++) mystery(x, 0 ,i);
  print(x, 8);                          // line (c)
  m2(a, b);    cout << a << b << endl; // line (d)
  m2(y[3], 7);       print(y, 6);       // line (e)
}
```

(a) What is the output at line (a)?

Answer:

```
0 1 2
```

(b) What is the output at line (b)?

Answer:

```
0 2 2 3 4
```

(c) What is the output at line (c)?

Answer:

```
7 2 2 3 4 5 6 7
```

(d) What is the output at line (d)?

Answer:

```
44
```

(e) What is the output at line (e)?

Answer:

```
0 1 2 7 4 5
```

Problem 123 Write a function called *doubleDigit* that makes each digit of an input parameter repeat twice. For example, a program that uses the function *doubleDigit* follows.

```
int main() {
   cout << doubleDigit(9)   << endl;          // prints 99
   cout << doubleDigit(81)  << endl;          // prints 8811
   cout << doubleDigit(243) << endl;          // prints 224433
   cout << doubleDigit(244) << endl;          // prints 224444
   return 0;
}
```

Answer:

```
int doubleDigit(int n) {
   if (n < 10) return n * 11;
   return 100 * doubleDigit(n / 10) + doubleDigit(n % 10);
}
```

Problem 124 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 1000 single digit integers. It then outputs the digit or digits that appears least often.

For example, if the user enters $3, 1, 4, 1, 5, 9, \ldots, 9, 8$ where 0 appears 93 times, 1 appears 116 times, 2 appears 103 times, 3 appears 103 times, 4 appears 93 times, 5 appears 97 times, 6 appears 94 times, 7 appears 95 times, 8 appears 101 times, 9 appears 105 times the output would be:

```
The digits 0 and 4 are least frequent.
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int count[10];
   int x;
   for (int c = 0; c < 10; c++)
      count [c] = 0;

   cout << "Enter 1000 single digit integers: ";
   for (int c = 1; c <= 1000; c++) {
      cin >> x;
      count[x]++;
   }

   int min = count[0];
   for (int c = 1; c <= 9; c++) {
      if (count[c] < min) min = count[c];
   }

   bool found = false;
   cout << "The digits ";
   for (int c = 0; c <= 9; c++) {
      if (count[c] == min) {
         if (found) cout << "and ";
         cout << c << " ";
      }
```

```
      found = true;
   }
   cout << "are least frequent.\n";
   return 0;
}
```

Problem 125 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **detectAge** which returns a user's age (by asking for input and rejecting negative values).

Answer:

```
int detectAge()
```

(b) A function called **sortString** that sorts an array of strings into alphabetical order.

Answer:

```
void sortString(string a[], int cap)
```

(c) A function called **sort4** that sorts 4 integer parameters into increasing order.

Answer:

```
void sort4(int &a, int &b, int &c, int &d)
```

(d) A function called **printCode** that prints the ASCII code for a character.

Answer:

```
void printCode(char x)
```

(e) A function called **delete7** which alters an integer parameter by deleting every occurrence of the digit 7.

Answer:

```
void delete7(int &x)
```

Problem 126 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int x[][4], int a, int b, int k) {
  for (int r = a; r <= b; r++) for (int c = a; c <= b; c++)
      x[r][c] = k;
}

void print(int x[][4], int s) {
  for (int r = 0; r < s; r++) {
     for (int c = 0; c < s; c++) cout << x[r][c];
     cout << endl;
  }
  cout << endl;
}

int main() {
  int x[4][4] = {{0,0,0,0}, {0,0,0,0}, {0,0,0,0}, {0,0,0,0}};
  mystery(x, 3, 2, 1); print(x, 4);  // line (a)
  mystery(x, 0, 1, 2); print(x, 4);  // line (b)
  mystery(x, 1, 2, 3); print(x, 4);  // line (c)
  mystery(x, 1, 3, 4); print(x, 4);  // line (d)
  mystery(x, 0, 3, 5); print(x, 2);  // line (e)
  return 0;
}
```

(a) What is the output at line (a)?

Answer:

```
0000
0000
0000
0000
```

(b) What is the output at line (b)?

Answer:

```
2200
2200
0000
0000
```

(c) What is the output at line (c)?

Answer:

```
2200
2330
0330
0000
```

(d) What is the output at line (d)?

Answer:

```
2200
2444
0444
0444
```

(e) What is the output at line (e)?

Answer:

```
55
55
```

Problem 127 Write a function called *cutNine* that prints the part of a number that follows its last 9 digit. (If there is no 9 digit, the whole number is printed. If the last digit is a 9, nothing is printed.)

For example, a program that uses the function *cutNine* follows.

```
int main() {
   cutNine(770);          cout << endl;          // prints 770
   cutNine(135792468);    cout << endl;          // prints 2468
   cutNine(1991991);      cout << endl;          // prints 1
   cutNine(1991999);      cout << endl;          // prints
   return 0;
}
```

Answer:

```
void cutNine(int n) {
  if (n == 0 || n % 10 == 9) return;
  cutNine(n/10);
  cout << n % 10;
}
```

Problem 128 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 1000 single digit integers. It then outputs the number of times that each digit was seen.

For example, if the user enters $3, 1, 4, 1, 5, 9, \ldots, 9, 8$ where 0 appears 93 times, 1 appears 116 times, ..., 9 appears 105 times, the output would be:

```
0 count 93, 1 count 116, 2 count 103, 3 count 103, 4 count 93,
5 count 97, 6 count 94, 7 count 95, 8 count 101, 9 count 105.
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int count[10];
   int x;
   for (int c = 0; c < 10; c++)
      count [c] = 0;

   cout << "Enter 1000 single digit integers: ";
   for (int c = 1; c <= 1000; c++) {
      cin >> x;
      count[x]++;
   }

   for (int c = 0; c < 10; c++) {
      cout << c << " count " << count[c];
      if (c % 5 < 4) cout << ", ";
      else if (c == 4) cout << ",\n";
      else cout << ".\n";
   }
   return 0;
}
```

Problem 129 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **add3** which returns the sum of three double parameters.

Answer:

```
double add3 (double a, double b, double c)
```

(b) A function called **reverseIt** that returns the number obtained by reversing the digits of an integer parameter.

Answer:

```
int reverseIt (int x)
```

(c) A function called **randomArray** that sets the values in an array of doubles to have random values.

Answer:

```
void randomArray (double arr [], int capacity)
```

(d) A function called **add5** that adds 5 to every entry in a 2-dimensional array each of whose rows has 35 columns.

Answer:

```
void add5 (int arr [][35], int rows, int columns)
```

(e) A function called **deleteX** which alters a string parameter by deleting every occurence of the letter X.
Answer:

```
void deleteX (string &str)
```

Problem 130 Consider the following C++ program.

```
#include <iostream>
using namespace std;

string fun(string x[], int y) {
   if (y <= 0) return x[1];
   if (y == 1) return x[0] + x[2];
   if (y == 2) return "illegal";
   if (y <= 4) return " 4";
   return "X" + fun(x, y - 6);

}

int main() {
  string array[3] = { "1", "2", "3"};
  cout << fun(array,0) << endl;            // line a
  cout << fun(array,1) << endl;            // line b
  cout << fun(array,2) << endl;            // line c
  cout << fun(array,4) << endl;            // line d
  cout << fun(array,12) << endl;           // line e
  return 0;
}
```

What is the output from the program at each of the following lines:
(a) line a:

```
2
```

(b) line b:

```
13
```

(c) line c:

```
illegal
```

(d) line d:

```
4
```

(e) line e:

```
XX2
```

Problem 131 Write a function called *makeOne* that returns the result of turning every odd valued digit in an integer parameter to a 1.

For example, a program that uses the function *makeOne* follows.

```
int main() {
   cout << makeOne(770)   << endl;    // prints 110
   cout << makeOne(13579) << endl;    // prints 11111
   cout << makeOne(1000)  << endl;    // prints 1000
   return 0;
}
```

Answer:

```
int makeOne (int x)
{
        if (x < 10 && x % 2 == 1) return 1;
        if (x < 10) return x;
        return 10 * makeOne (x / 10) + makeOne (x % 10);
}
```

Problem 132 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 3 positive integers. It then outputs the least frequently encountered digit or digits in those 3 numbers.

For example, if the user enters the integers 123, 45678, and 200 the program should output 9 which occurs less often than any other digit in these numbers.

Answer:

```
#include <iostream>
using namespace std;

void getDigits (int n, int count []) {
   while (n > 0) {
      int digit = n % 10;
      count [digit]++;
      n /= 10;
   } //while
}

int main () {
   int n1, n2, n3;
   cout << "Enter three positive integers: ";
   cin >> n1 >> n2 >> n3;

   int count [10];
   for (int i = 0; i < 10; i++)
      count [i] = 0;

   getDigits (n1, count);
   getDigits (n2, count);
   getDigits (n3, count);

   int min = count[0];
   for (int i = 1; i < 10; i++)
        if (count[i] < min) min = count[i];

   cout << "The following digits occur least often:" << endl;
   for (int i = 0; i < 10; i++)
        if (count[i] == min) cout << i << endl;

   return 0;
}
```

Problem 133 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **add3** which returns the sum of three integer parameters.

Answer:

```
int add3(int a, int b, int c)
```

(b) A function called **reverseString** that returns the reverse of a string.

Answer:

```
string reverseString(string str)
```

(c) A function called **randomArray** that sets the values in an array of integers to have random values.

Answer:

```
void randomArray (int arr [], int capacity)
```

(d) A function called **add3** that adds 3 to every entry in a 2-dimensional array each of whose rows has 25 columns.

Answer:

```
void add3 (int arr [][25], int rows, int columns)
```

(e) A function called **deleteX** which alters a string parameter by deleting every occurence of the letter X.

Answer:

```
void deleteX (string &str)
```

Problem 134 Consider the following C++ program.

```
#include <iostream>
using namespace std;

string fun(string x[], int y) {
   if (y <= 0) return x[0];
   if (y == 1) return x[1] + x[2];
   if (y == 2) return "illegal";
   if (y <= 4) return " <= 4";
   return "X" + fun(x, y - 5);

}

int main() {
  string array[3] = { "1", "2", "3"};
  cout << fun(array,0) << endl;            // line a
  cout << fun(array,1) << endl;            // line b
  cout << fun(array,2) << endl;            // line c
  cout << fun(array,4) << endl;            // line d
  cout << fun(array,12) << endl;           // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
1
```

(b) line b:

```
23
```

(c) line c:

```
Illegal
```

(d) line d:

```
<= 4
```

(e) line e:

```
XXillegal
```

Problem 135 Write a function called *makeOne* that returns the result of turning every non-zero digit in an integer parameter to a 1.

For example, a program that uses the function *makeOne* follows.

```
int main() {
   cout << makeOne(770)   << endl;    // prints 110
   cout << makeOne(13579) << endl;    // prints 11111
   cout << makeOne(1000)  << endl;    // prints 1000
   return 0;
}
```

Answer:

```
int makeOne(int x)
{
        if (x == 0) return 0;
        if (x < 10) return 1;
        return 10 * makeOne (x / 10) + makeOne (x % 10);
}
```

Problem 136 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 3 positive integers. It then outputs the most frequently encountered digit or digits in those 3 numbers.

For example, if the user enters the integers 737, 13579, and 246 the program should output 7 which occurs more often than any other digit in these numbers.

Answer:

```
#include <iostream>
using namespace std;

void getDigits (int n, int count []) {
   while (n > 0) {
      int digit = n % 10;
      count [digit]++;
      n /= 10;
   } //while
}

int main () {
   int n1, n2, n3;
```

```
   cout << "Enter three positive integers: ";
   cin >> n1 >> n2 >> n3;

   int count [10];
   for (int i = 0; i < 10; i++)
      count [i] = 0;

   getDigits (n1, count);
   getDigits (n2, count);
   getDigits (n3, count);

   int max = count[0];
   for (int i = 1; i < 10; i++)
        if (count[i] > max) max = count[i];

   cout << "The following digits occur most often:" << endl;
        for (int i = 0; i < 10; i++)
           if (count[i] == max) cout << i << endl;

   return 0;
}
```

Problem 137 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {3,1,4,1}, b[5] = {2,7,1,8,1}, i = 3, j = 5, k = 4;
   int x[2][2] = {{0,1},{3,2}};
   cout << max(x, 2 , 2); // outputs:  3
   printArray(a, 4);  // outputs:  3,1,4,1
   reverse(a, 0, 3);  // changes a to 1,4,1,3
   sort1(b, 5);
   printArray(b, 5);  // outputs:  1,1,2,7,8
   sort2(i, j, k);
   cout << i << j << k << endl;  // outputs: 345
   return 0;
}
```

(a) Title line for **max**

Answer:

```
int max(int x[][2], int a, int b)
```

(b) Title line for **printArray**

Answer:

```
void printArray(int array[], int cap)
```

(c) Title line for **reverse**

Answer:

```
void reverse(int array[], int from, int to)
```

(d) Title line for **sort1**

Answer:

```
void sort1(int array[], int n)
```

(e) Title line for **sort2**

Answer:

```
void sort2(int &a, int &b, int &c)
```

Problem 138 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void rec(int a[], int start, int stop) {
   if (stop <= start) return;
   a[start] = a[stop];
   rec(a, start + 1, stop -1);
}

void printA(int a[], int cap) {
  for (int c = cap - 1; c >= 0; c--) cout << a[c] << " ";
  cout << endl;
}

int main() {
  int x[6] = {0, 1, 2, 3, 4, 5};

  printA(x, 6);                  // line (a)
  printA(x, 4);                  // line (b)
  rec(x, 3, 3); printA(x, 4);    // line (c)
  rec(x, 3, 4); printA(x, 6);    // line (d)
  rec(x, 0, 5); printA(x, 6);    // line (e)

  return 0;
}
```

What is the output at each of the following lines?

(a) line (a)

```
5 4 3 2 1 0
```

(b) line (b)

```
3 2 1 0
```

(c) line (c)

```
3 2 1 0
```

(d) line (d)

```
5 4 4 2 1 0
```

(e) line (e)

```
5 4 4 4 4 5
```

Problem 139 Write a function called *maxMid* that determines the maximum value in the middle column of a 2-dimensional array of numbers of type double. (You should assume that the 2-dimensional array has an odd number of columns.)

For example, a program that uses the function *maxMid* follows. Your function must complete this program.

```
int main() {
   double x[4][5] = {{0,1,2,3,4}, {1,2,3,4,5}, {2,3,4,5,6}, {5,6,7,8,9}};
   cout << maxMid(x, 4, 5) << endl;  // prints 7.0
   return 0;
}
```

Answer:

```
double maxMid(double x[][5], int rows, int cols) {
   double ans = x[0][cols / 2];
   for (int i = 0; i < rows; i++)
      if ((x[i][cols / 2] > ans) ans = x[i][cols / 2];
   return ans;
}
```

Problem 140 Write a complete C++ program that does the following. (In your program, you do not need to check whether the user enters legal input.)

1. It asks the user to enter a positive integer n that is at most 100.

2. The program reads n single digit integers entered by the user. (A single digit integer is an integer n with $0 \leq n \leq 9$.)

3. The program prints a list of all single digit integers that were not entered at all by the user.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):    11
Enter 11 single digit integers:
1 1 7 3 3 2 0 3 7 7 7
The following were not entered: 4 5 6 8 9
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int n, c, x, count[10];
   for (int c = 0; c <= 9; c++) count[c] = 0;

   cout << "Enter a positive integer (at most 100): ";
   cin >> n;
   cout << "Enter " << n << " single digit integers: ";

   for (int c = 0; c < n; c++) {
      cin >> x;
      count[x]++;
   }
   cout << "The following were not entered:";
   for (int c = 0; c <= 9; c++)
      if (count[c] == 0) cout << " " << c;
   cout << endl;

   return 0;
}
```

Problem 141 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **welcome** which prints the word "Hello" to the screen.

Answer:

```
void welcome()
```

(b) A function called **addTwo** that adds 2 to every entry in an array of integers.

Answer:

```
void addTwo(int array[], int cap)
```

(c) A function called **randomTruth** which determines and returns a random true/false result.

Answer:

```
bool randomTruth()
```

(d) A function called **numberPrimes** which returns the number of prime numbers that lie between a specified pair of input values.

Answer:

```
int numberPrimes(int a, int b)
```

(e) A function called **biggerAverage** which determines which of two arrays of integers has the bigger average. It should return the value of this bigger average.

Answer:

```
double biggerAverage(int array1[], int cap1, int array2[], int cap2)
```

Problem 142 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   x = y + 1;
   y = x + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  fun(x, 8);                    // line a
  fun(x, y);                    // line b
  fun(y, x);                    // line c
  fun(y, x);                    // line d
  cout << fun(x, 3) << endl;    // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

```
910
```

(b) line b:

```
12
```

(c) line c:

```
23
```

(d) line d:

```
23
```

(e) line e:

```
45
20
```

Problem 143 Write a function called *alternates* that prints every second digit of an integer parameter, starting from the right.

For example, a program that uses the function *alternates* follows.

```
int main() {
   alternates(10);  cout << endl;     // prints 0
   alternates(123456); cout << endl;  // prints 642
   alternates(1000); cout << endl;    // prints 00
   return 0;
}
```

Answer:

```
void alternates(int n) {
   cout << n % 10;
   if (n >= 100) alternates(n/100);
}
```

An alternative solution that does not use recursion follows:

```
void alternates (int x) {
   while (x > 0) {
      cout << x % 10;
      x /= 100;
   }
}
```

Problem 144 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer that is between 1 and 26.

2. The program reads a value n entered by the user. If the value is not legal, the program exits.

3. The program prints an $n \times n$ pattern of characters, in which the top left character is an 'A'. The top left 2×2 block is completed by three 'B' characters. The top left 3×3 block is completed by five 'C' characters, and so on.

For example, if the user enters 5 for n the program should print the following picture.

```
ABCDE
BBCDE
CCCDE
DDDDE
EEEEE
```

Answer:

```
#include <iostream>
using namespace std;
```

```
int main() {
   int r, c, x, n;
   char pic[26][26];

   cout << "Enter an integer between 1 and 26: ";
   cin >> n;
   if (n < 1 || n > 26) exit(1);

   for (x = n - 1; x >= 0; x--)
      for (r = 0; r <= x; r++)
         for (c = 0; c <= x; c++) pic[r][c] = 'A' + x;

   for (r = 0; r <= n - 1; r++) {
      for (c = 0; c <= n - 1; c++) cout << pic[r][c];
      cout << endl;
   }

   return 0;
}
```

Problem 145 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **firstDigit** which returns the first digit of an integer.

Answer:

```
int firstDigit(int x)
```

(b) A function called **sqrt** that returns the square root of a double precision parameter.

Answer:

```
double sqrt(double x)
```

(c) A function called **oddString** which returns a string made up of the characters in odd position of an input string.

Answer:

```
string oddString(string s)
```

(d) A function called **randomWord** which is to create and return a random word.

Answer:

```
string randomWord()
```

(e) A function called **sort** which is to sort an array of strings into alphabetical order.

Answer:

```
void sort(string data[], int cap)
```

Problem 146 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 10) return n;
   if (n < 100) return n/10;
   return 10 * recursive(n / 100) + n % 10;
```

```
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

Answer:

```
5
```

(b) The user enters 16 for x.

Answer:

```
1
```

(c) The user enters 123 for x.

Answer:

```
13
```

(d) The user enters 1234 for x.

Answer:

```
14
```

(e) The user enters 19683 for x.

Answer:

```
163
```

Problem 147 Write a function called *evens* that deletes all odd digits from a positive integer parameter. For example, a program that uses the function *evens* follows.

```
int main() {
   cout << evens(16) << endl;       // prints 6
   cout << evens(666) << endl;      // prints 666
   cout << evens(777) << endl;      // prints 0
   return 0;
}
```

Answer:

```
int evens(int n){
   if (n % 2 == 1) return evens(n / 10);
   if (n < 10) return n;
   return 10 * evens(n / 10) + n % 10;
}
```

Problem 148 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in a 2-dimensional array with n rows and n columns of integers entered by the user.
3. The program prints out the average of the entries for each column of the array.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     3
Enter 3 rows of 3 integers:
 3 -1    4
10 30 -100
 2 -2   99
The averages of the 3 columns are: 5.0 9.0 1.0
```

Answer:

```
#include <iostream>
using namespace std;

int main ()
{
        int num;
        int arr [100] [100];
        cout << "Give a number that's at most 100: ";
        cin >> num;

        cout << "Give me " << num << " rows of " << num << " integers: ";
        for (int r = 0; r < num; r++)
                for (int c = 0; c < num; c++)
                        cin >> arr [r] [c];

        cout << "The averages of the " << num << " columns are:";

        for (int c = 0; c < num; c++)
        {
                int colSum = 0;
                for (int r = 0; r < num; r++)
                    colSum += arr[r][c];
                cout << ((double) colSum) / num << " ";
        } //for

        cout << endl << endl;

        return 0;
} //main
```

Problem 149 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Include declarations for any variable that you use.

(i) Print the remainder when 101 is divided by 17 to the file *out.txt*. **Answer:**

```
ofstream out("out.txt");
out << 101 % 17;
```

(ii) Print a random lower case letter to the screen. (The random letter should be determined by using an appropriate C++ function.) **Answer:**

```
cout << (char) ('a' + rand() % 26);
```

(iii) Read a line of text from the user and print the word *Yes* if it contains the character 7. **Answer:**

```
string input;
getline(cin, input);
if (input.find("7") >= 0) cout << "Yes";
```

(iv) Print the middle character of the string s. (Assume that the string has odd length.) **Answer:**

```
cout << s[s.length() / 2];
```

(v) Swap the values of integer variables called x and y. **Answer:**

```
int temp = y;
y = x;
x = temp;
```

Problem 150 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 10) return n;
   return 100 * recursive(n / 100) + 11* (n % 10);
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

Answer:

```
5
```

(b) The user enters -10 for x.

Answer:

```
-10
```

(c) The user enters 65 for x.

Answer:

```
55
```

(d) The user enters 123 for x.

Answer:

```
133
```

(e) The user enters 19683 for x.

Answer:

```
16633
```

Problem 151 Write a function called *twoPart* that returns the largest power of 2 that divides a positive integer parameter.

For example, a program that uses the function *twoPart* follows.

```
int main() {
   cout << twoPart(16) << endl;   // prints 16
   cout << twoPart(666) << endl;   // prints 2
   cout << twoPart(777) << endl;   // prints 1
   return 0;
}
```

Answer:

```
   int twoPart(int x) {
      if (x % 2 == 1) return 1;
      return 2 * twoPart (x / 2);
   }
```

Problem 152 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in a 2-dimensional array with n rows and n columns of integers entered by the user.
3. The program prints out the maximum entry found for each row of the array.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     3
Enter 3 rows of 3 integers:
3 -1 4
10 30 -100
0 0 0
The maximum entries in the 3 rows are: 4 30 0
```

Answer:

```
#include <iostream>
using namespace std;

int main ()
{
        int num;
        int arr [100] [100];
        cout << "Give a number that's at most 100: ";
        cin >> num;

        cout << "Give me " << num << " rows of " << num << " integers: ";
        for (int r = 0; r < num; r++)
                for (int c = 0; c < num; c++)
                        cin >> arr [r] [c];

        cout << "The maximum entries in the " << num << " rows are:";

        int max;
        for (int i = 0; i < num; i++)
        {
                max = arr [i] [0];
                for (int j = 0; j < num; j++)
                        if (max < arr [i] [j])
                                max = arr [i] [j];
                cout << max << " ";
        } //for

        cout << endl << endl;
```

```
        return 0;
} //main
```

Problem 153 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  int x[10], z[10][10], r, c;
  string s;
```

(i) Print the remainder when r is divided by c.

```
cout << r % c;
```

(ii) Set r to be a random integer between 1 and 10. (The random integer should be determined by an appropriate C++ function.)

```
r = rand() % 10 + 1;
```

(iii) Print the sum of all 10 entries of the array x.

```
int sum = 0;
for (int i = 0; i < 10; i++) sum += x[i];
cout << sum;
```

(iv) Print the last character of the string s.

```
cout << s[s.size() - 1];
```

(v) Swap row number 0 with row number 4 in the 2-dimensional array z.

```
for (int i = 0; i < 10; i++) {
   int temp = z[0][i];
   z[0][i] = z[4][i];
   z[4][i] = temp;
}
```

Problem 154 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void x1(int a[][6], int n) {
   for (int i = 0; i < 5; i++) cout << a[n][i];
   cout << endl;
}

void x2(int b[][6], int n) {
   for (int i = 0; i < n; i++)
      cout << b[i][i] << " ";
   x1(b, n);
}

main() {
```

```
  int x[6][6], a[6][6], b[6][6];
  for (int i = 0; i < 6; i++) for (int j = 0; j < 6; j++) {
     x[i][j] = i + j;
     a[i][j] = i * j;
     b[i][j] = (i + 1) / (j + 1);
  }
  cout << "Part a: " << x[5][4] << endl;
  cout << "Part b: " << a[5][4] << endl;
  cout << "Part c: "; x1(x, 5);
  cout << "Part d: "; x2(x, 5);
  cout << "Part e: "; x2(b, 3);
  return 0;
}
```

Complete the line of output that begins:

Answer:

```
Part a: 9
Part b: 20
Part c: 56789
Part d: 0 2 4 6 8 56789
Part e: 1 1 1 42110
```

Problem 155 Write a function called *sixCount* that returns a count of the number of digits that are equal to 6 in its positive integer parameter.

For example, a program that uses the function *sixCount* follows.

```
int main() {
   cout << sixCount(16) << endl;   // prints 1
   cout << sixCount(666) << endl;   // prints 3
   cout << sixCount(777) << endl;   // prints 0
   return 0;
}
```

Answer:

```
int sixCount(int x) {
   if (x == 6) return 1;
   if (x < 10) return 0;
   return sixCount(x % 10) + sixCount(x / 10);
}
```

Problem 156 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in an array n integers entered by the user.
3. The program prints the negative entries from the array, in order.
4. The program prints the positive entries from the array in reverse order.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     8
Enter 8 integers:  3 -1 4 -10 17 18 19 -11
-1 -10 -11
19 18 17 4 3
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int x[100];
   int count;

   cout << "Enter a positive integer (at most 100):  ";
   cin >> count;
   cout << "Enter " << count << " integers:  ";
   for (int i = 0; i < count; i++) cin >> x[i];

   for (int i = 0; i < count; i++)
      if (x[i] < 0) cout << x[i] << " ";
   cout << endl;
   for (int i = count - 1; i >= 0; i--)
      if (x[i] > 0) cout << x[i] << " ";
   cout << endl;
   return 0;
}
```

Problem 157 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  int x[10], y[10], z[10][10], r, c;
```

(i) Read 10 integers into the array x.

Answer:

```
for (c = 0; c <= 9; c++) cin >> x[c];
```

(ii) Set all the entries of the array z so that the entry in row r and column c stores the product of r and c.

Answer:

```
for (r = 0; r <= 9; r ++) for (c = 0; c <= 9; c++)
   z[r][c] = r * c;
```

(iii) Print the smallest value in the array x.

Answer:

```
int min = x[0];
for (c = 1; c <= 9; c++) if (x[c] < min) min = x[c];
cout << min;
```

(iv) Print the word *Divides* if r divides exactly into c otherwise do nothing.

Answer:

```
if (c % r == 0) cout << "Divides";
```

(v) Swap each entry of the array x with the correpsonding entry of array y.

Answer:

```
for (c = 0; c <= 9; c++) {
   int temp = x[c];
   x[c] = y[c];
   y[c] = temp;
}
```

Problem 158 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 100) return n%10;
   return 10 * recursive(n / 100) + n % 10;
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters -10 for x.

Answer: 0

(b) The user enters 5 for x.

Answer: 5

(c) The user enters 55 for x.

Answer: 5

(d) The user enters 123 for x.

Answer: 13

(e) The user enters 19683 for x.

Answer: 163

Problem 159 Write a function called *toTen* that calculates how many entries of an array need to be added to make a sum of 10 or more. (Start adding from index 0.)

For example, a program that uses the function *toTen* follows.

```
int main() {
   int x[8] = {5, 3, 1, 6, 10, 1, -30, -100};
   cout << toTen(x, 8) << endl;
   return 0;
}
```

The output from this program would be 4, because the sum of the first 4 entries $5 + 3 + 1 + 6$ is the first sum that exceeds 10.

Answer:

The following function returns an answer of -1 in case no sum of entries in the array reaches a value of 10. Exam solutions are not required to deal with this possibility.

```
int toTen(int x[], int c) {
  int sum = x[0];
  int col = 1;
  while (sum < 10 && col < c) {
    sum = sum + x[col];
    col ++;
  }
  if (sum < 10) return -1;
  return col;
}
```

Problem 160 Write a complete C++ program that does the following.

1. It asks the user to enter their name as a string *name*.
2. The program reads the name entered by the user.
3. The program converts all letters in the name to uppercase and prints the name.
4. The program prints the uppercase characters of the name in reverse.

For example, the following represents one run of the program.

```
What is your name:     Freddy
FREDDY
YDDERF
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   string name;
   cout << "What is your name: ";
   getline(cin, name);
   for (int i = 0; i < name.size(); i++)
      name[i] = toupper(name[i]);
   cout << name << endl;
   for (int i = name.size() - 1; i >= 0; i--)
     cout << name[i];
   cout << endl;
   return 0;
}
```

Problem 161 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **sumDigits** which returns the sum of the digits of an integer.

Answer:

```
int sumDigits(int x)
```

(b) A function called **isSmall** that returns an answer of true if a double precision parameter has a value between 0 and 0.001. (It returns false otherwise.)

Answer:

```
bool isSmall(double x)
```

(c) A function called **randomLetter** which generates and returns a random letter of the alphabet. (The output is to be a single character between 'A' and 'Z'.)

Answer:

```
char randomLetter()
```

(d) A function called **sort3** which is to change a collection of three input values so that they appear in increasing order.

Answer:

```
void sort3(int &x, int &y, int &z)
```

(e) A function called **total** which is to determine the sum of all the entries in an array.

Answer:

```
int total(int x[], int capacity)
```

Problem 162 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 10) return n;
   return n % 10 - recursive(n/10);
}

main() {
  int x;
  cout << "Enter a positive integer: ";
  cin >> x;
  if (x <= 0) cout << "Error" << endl;
  else cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 0 for x.

Answer:

```
Error
```

(b) The user enters 5 for x.

Answer:

```
5
```

(c) The user enters 55 for x.

Answer:

```
0
```

(d) The user enters 555 for x.

Answer:

```
5
```

(e) The user enters 19683 for x.

Answer:

```
-7
```

Problem 163 Write a function called *quadratic* that calculates the value of a quadratic function $ax^2 + bx + c$. For example, a program that uses the function *quadratic* follows.

```
int main() {
   double a = 1.0, b = 2.2, c = 1.21, x = 0.1;
   cout << quadratic(a, b, c, x) << endl;
   return 0;
}
```

Answer:

```
double quadratic(double a, double b, double c, double x) {
   return c + b * x + a * x * x;
}
```

Problem 164 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, n.

2. The program reads a value entered by the user. If the value is not positive, the program should terminate.

3. The program should consider every number x between 1 and n and print out any value of x that divides exactly into n.

The printed values should all appear on a single line, separated by spaces.

For example, the following represents one run of the program. (The user chooses the number 28.)

```
Enter a positive integer:     28
1 2 4 7 14 28
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int n, x;

   cout << "Enter a positive integer value for n: ";
   cin >> n;
   if (n <= 0) exit(1);

   for (x = 1; x <= n; x++)
     if (n % x == 0) cout << x << " ";

   cout << endl;
   return 0;
}
```

Problem 165 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **sum** which returns the sum of 4 double precision values.

Answer:

```
double sum(double a, double b, double c, double d)
```

(b) A function called **midDigit** that is used to return the middle digit of an integer.

Answer:

```
int midDigit(int x)
```

(c) A function called **isPositive** which is to return an answer of true if the sum of the entries of an array of double precision data is positive (and return false otherwise).

Answer:

```
bool isPositive(double x[], int capacity)
```

(d) A function called **average2DArray** which is to print (to cout) the average of the entries in a 2-dimensional array (the array stores integers and has 10 rows and 15 columns).

Answer:

```
void average2DArray(int array[][15], int rows, int cols)
```

(e) A function called **makeZero** which is to use two integer input variables and change their values to zero. (After the function ends, the input variables must be zero.)

Answer:

```
void makeZero(int &x, int &y)
```

Problem 166 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int n) {
   cout << n % 100;
   if (n < 1000) return;
   mystery(n/10);
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  mystery(x);
  cout << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

Answer: 5

(b) The user enters 512 for x.

Answer: 12

(c) The user enters 4370 for x.

Answer: 7037

(d) The user enters 175560 for x.

Answer: 60565575

Problem 167 Write a function called $sum2D$ that returns the sum of all elements in a 2-dimensional array that has 4 columns of integer entries.

For example, a program that uses the function $sum2D$ follows.

```
int main() {
   int array[3][4] = {{1,2,3,4},{1,2,3,4},{1,2,3,4}};
   cout << sum2D(array, 3, 4) << endl;
   return 0;
}
```

The input values 3 and 4 specify the number of rows and columns in the array. The program should print an answer of 30 (since this is the sum of $1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3,$ and 4).

Answer:

```
int sum2D(int a[][4], int r, int c) {
   int ans = 0;
```

```
   for (int row = 0; row < r; row++)
     for (int col = 0; col < c; col++)
       ans += a[row][col];
   return ans;
}
```

Problem 168 Write a complete C++ program that does the following.

1. It asks the user to enter a 5-digit integer value, n.
2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.
3. The program calulates and stores the 5 individual digits of n.
4. The program outputs a "bar code" made of 5 lines of stars that represent the digits of the number n.

For example, the following represents one run of the program. (The user chooses the number 16384.)

```
Enter a 5 digit integer:     16384
*
******
***
********
****
```

Answer:

```
#include <iostream>
using namespace std;

int bar(int l) {
   for (int c = 0; c < l; c++) cout << "*";
   cout << endl;
}

int main() {
   int i, n;
   int digit[5];

   cout << "Enter a 5 digit integer: ";
   cin >> n;
   if (n < 10000 || n > 99999) exit(0);

   for (i = 0; i < 5; i++) {
      digit[i] = n % 10;
      n = n / 10;
   }

   for (i = 4; i >= 0; i--) bar(digit[i]);
   return 0;
}
```

Here is an alternative solution that is shorter, but makes use of recursion:

```
#include <iostream>
using namespace std;

void bars(int n) {
   if (n == 0) return;
   bars(n/10);
   for (int c = 0; c < n % 10; c++) cout << "*";
   cout << endl;
```

```
}

int main() {
   int n;
   cout << "Enter a 5 digit integer: ";
   cin >> n;
   if (n < 10000 || n > 99999) exit(0);
   bars(n);
   return 0;
}
```

Problem 169 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **lastDigit** that is used to find the last digit of an integer.

Answer:

```
int lastDigit(int x)
```

(b) A function called **average** which determines the average of 3 integer values.

Answer:

```
double average(int x, int y, int z)
```

(c) A function called **largest** which is used to find the largest value in an array of double precision data.

Answer:

```
double largest(double array[], int cap)
```

(d) A function called **print2DArray** which is to print out all of the data in a 2-dimensional array of integers(the array has 100 columns).

Answer:

```
void print2DArray(int array[][100], int rows, int cols)
```

(e) A function called **sort** which is to sort an array of strings into alphabetical order.

Answer:

```
void sort(string array[], int cap)
```

Problem 170 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int data[], int p, int q) {
  data[p] = data[q];
  data[q] = data[p];
}

void m2(int p, int q) {
  int temp = p;
  q = p;
  p = temp;
}

void print(int data[], int p) {
```

```
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int scores[8] = {3, 1, 4, 1, 5, 9, 2, 6};
  int quiz[7] = {0, 1, 2, 3, 4, 5, 6};
  print(scores, 3);
  print(quiz, 4);
  mystery(scores, 1, 2);
  print(scores, 5);
  for (int i = 0; i < 3; i++)
     m2(quiz[i], quiz[i+ 1]);
  print(quiz, 6);
}
```

What is the output from the program?

Answer:

```
3 1 4
0 1 2 3
3 4 4 1 5
0 1 2 3 4 5
```

Problem 171 Write a function called *countChange* that uses four parameters q, d, n, and p and converts the value of q auarters, d dimes, n nickels, and p cents into dollars.

For example, a program that uses the function *countChange* follows.

```
int main() {
   int q = 10, d = 5, n = 1, p = 2;
   double x = countChange(q, d, n, p);
   cout << "You have $" << x << endl;
}
```

It should print:

```
You have $3.07
```

Answer:

```
double countChange(int quarters, int dimes, int nickels, int pennies) {
   return quarters * 0.25 + dimes * 0.1 + nickels * 0.05 + pennies * 0.01;
}
```

Problem 172 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, r that is at most 100.

2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.

3. The program reads and stores r integers from the user and then prints a pattern of r rows of stars, the lengths of which are the other integers entered by the user.

For example, the following represents one run of the program.

```
How many rows? 4
Enter 4 row lengths:  2 7 1 5
**
*******
*
*****
```

Answer:

```
#include <iostream>
using namespace std;
int main() {
   int arr[100];
   int r, i, j;

   cout << "How many rows? ";
   cin >> r;
   if (r < 1 || r > 100) exit(1);

   cout << "Enter " << r << " row lengths: ";
   for (i = 0; i < r; i++) cin >> arr[i];

   for (i = 0; i < r; i++) {
      for (j = 0; j < arr[i]; j++) cout << "*";
      cout << endl;
   }

   return 0;
}
```

Problem 173 Write a C++ program that asks a user how many times it should say hello and then says hello the required number of times. For example, a run of the program might produce the following output:

```
How many hellos do you want: 6
Hello Hello Hello Hello Hello Hello
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int n;
   cout << " How many hellos do you want: ";
   cin >> n;
   for (int c = 1; c <= n; c++) cout << "Hello ";
   cout << endl;
   return 0;
}
```

Problem 174 Two numbers are considered as very different if they differ by more than 10. Write a C++ function called areVeryDifferent that determines whether two integers are very different.

For example, your function could be used in the following program.

```
int main() {
   int x = 4, y = 10, z = -4;
   if (areVeryDifferent(x, y)) cout << "x and y are very different" << endl;
   if (areVeryDifferent(x, z)) cout << "x and z are very different" << endl;
   if (areVeryDifferent(y, z)) cout << "y and z are very different" << endl;
   return 0;
}
```

The output from this program would be:

```
y and z are very different
```

Answer:

```
bool areVeryDifferent(int x, int y) {
   if ((x - y) > 10 || (y - x) > 10) return true;
   return false;
}
```

Problem 175 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, x that is at most 100.
2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.
3. The program reads and stores x words from the user and then prints them in reverse order.

For example, the following represents one run of the program.

```
How many words? 5
Freddy and Max were absent
absent were Max and Freddy
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   string data[100];
   int n;
   cout << "How many words (between 1 and 100): ";
   cin >> n;
   if (n <= 0 || n > 100) exit(0);

   for (int c = 0; c < n; c++) cin >>data[c];
   for (int c = (n - 1); c >= 0; c--) cout << data[c] << " ";
   cout << endl;

   return 0;
}
```

Problem 176 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int data[], int p, int q) {
  data[p] = data[q] + data[p];
  data[q] = 0;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int scores[8] = {3, 1, 4, 1, 5, 9, 2, 6};
```

```
  int quiz[7] = {0, 1, 2, 3, 4, 5, 6};
  print(quiz, 7);
  print(scores, 8);
  mystery(scores, 3, 4);
  print(scores, 8);
  for (int i = 1; i < 7; i++)
     mystery(quiz, 0, i);
  print(quiz, 7);
}
```

What is the output from the program?

Answer:

```
0 1 2 3 4 5 6
3 1 4 1 5 9 2 6
3 1 4 6 0 9 2 6
21 0 0 0 0 0 0
```